Visual analysis for sports sensor data

Visualizing measurements to improve performance

Master's thesis

R. Nieuwenhuizen

Computer Graphics and Visualization
Department of Intelligent Systems
Faculty of Electrical Engineering, Mathematics and Computer Science
Delft University of Technology
Visual analysis for sports sensor data

Visualizing measurements to improve performance

by

R. Nieuwenhuizen

to obtain the degree of Master of Science in Computer Science at the Delft University of Technology, to be defended publicly on January 13, 2017 at 11:00.

Student number: 4080408
Project duration: January 4, 2016 – January 13, 2017
Thesis committee: Dr. A. Vilanova, TU Delft, supervisor
Dr. J. van Gemert, TU Delft
Dr. A. L. Schwab, TU Delft

An electronic version of this thesis is available at http://repository.tudelft.nl/.
Abstract

In sports, a lot of measurements are done, but most of the time the researchers lack a good analysis tool, and have to do everything by hand. In this work, we present a design for a visual analysis tool for time-varying data. The tool can be used to inspect and compare sports datasets, and to form hypotheses on what parameter values cause a good performance. This is done using a dimensionality reduction algorithm on the datasets, allowing the user to visually group similar datasets. Different visualizations are designed for scalar data, vector data, angle data and position data. The tool takes previously gathered data from different sports. For example sensor data from a measurement bike, which is created to measure cyclists performance on downhill roads; sensor data from an instrumented skate, which is created to measure speed skaters forces and posture; or movement data from baseball pitchers. This data is visualized in a comprehensible way, in order to be able to extract a ‘best way’ of descending, based on the differences between the faster and slower riders; the most efficient power distribution in speed skating, in terms of energy and performance; or the ‘fastest and least injury-prone’ trajectory of arm movement in baseball pitching.
Preface

This thesis, “Visual analysis for sports sensor data”, describes the development process of a visual analysis tool that can help sports scientists in understanding their data, which will hopefully lead to improvements in sports performances. It is the result of my 2016: a year of programming and thesis writing. It is also the final step in obtaining my Master of Science degree in Computer Science: Media and Knowledge Engineering at the Delft University of Technology.

The project started when I got engaged in another project, where a TU Delft student created a measurement bike for Team Giant Alpecin. Since I am very interested in sports, continuing on this project seemed like a good idea for my thesis. Eventually, due to restrictions in the amount of data that was available, my project expanded its reach to multiple sports: a good thing, since more people could be helped this way!

I would like to thank my supervisor Anna Vilanova, who kept me going through the year and even stood by me in the tough battle against the evil transformation matrices. My thanks also go to the sports researchers I have collaborated with. In our conversations you have given me nice insights in the sports you are involved in. I hope we can continue our collaboration to develop a state of the art tool for visual sports analysis.

Lastly, I want to thank you, reader, for taking the effort of reading my report. I hope you enjoy it to the end.

R. Nieuwenhuizen
Delft, December 2016
## Contents

1 Introduction .................................................. 1

2 Data and task analysis ........................................ 5
   2.1 Data description ........................................... 5
   2.2 Data abstraction ............................................ 8
   2.3 Task analysis ............................................... 8

3 Visualization design ........................................... 11
   3.1 Visualizations .............................................. 11
      3.1.1 Line plot .............................................. 12
      3.1.2 Vector plot ............................................ 12
      3.1.3 Movement plot ......................................... 13
      3.1.4 Trajectory plot ......................................... 15
      3.1.5 Event plot ............................................. 15
      3.1.6 Linked views .......................................... 15
   3.2 Identifying groups of datasets ............................ 18
      3.2.1 t-SNE embeddings ..................................... 18
      3.2.2 t-SNE algorithm ....................................... 19
      3.2.3 Embedding plot ......................................... 20
      3.2.4 Visualization of groups ................................. 21
   3.3 Selection of time periods .................................. 24
   3.4 Conclusion ................................................ 24

4 Evaluation ....................................................... 27
   4.1 Qualitative evaluation ...................................... 27
   4.2 Test results ................................................ 29

5 Conclusion and future work .................................... 33

A A guide to using the visual analysis tool ....................... 35

B Implementation details ......................................... 39

C List of variables ................................................. 43

D Transformations .................................................. 47

E Evaluation forms ................................................ 49

Bibliography ...................................................... 55
Introduction

In sports, a lot of measurements are being done, to be used in the media, to attract sponsors and entertain the public; by the athletes themselves, to improve their performance; and by organisations to improve their (medical) services by using the data in research.

Statistical analysis of these measurements is commonly used in sports. Given the large amounts of data that can be extracted in nearly every sports field, people are likely to get some interesting and significant findings on how a player or a team performs, or how people are attracted to a certain sport.

However, when you do not know what you are looking for and want to generate knowledge or hypotheses, statistical analysis is not always the adequate tool. An alternative option is visual analytics, where the data is visualized for the user in a comprehensible way, making it easier to spot points of interest in the data. These points of interest can then be used to form a hypothesis, which allows for more specific testing. Sports data visualization is a growing field, which becomes clear from a recent edition of the IEEE Computer Graphics and Applications journal [1], that is dedicated to this topic.

In sports research, small teams with minimal budgets often want to form and test hypotheses on how to improve a specific sports performance. For this group visual analytics is an interesting alternative. We describe three of these projects, where researchers try to answer the general question “what makes a winner?” This question can be simply answered by “hard work” or “talent”, but what the researchers want to know is what the focus of the “hard work” should be, so that coaches can efficiently schedule the time of the athlete. Also they want to know how to define a “talented” athlete: what skills define a talent for a certain sport, and how do talented athletes use their skills? In this report, we focus on three sports which all have their specific goals to approach these questions in a way that has never been done for their specific sports.

“Cornering and descending are often overlooked aspects of cycling. However, in racing, descents can sometimes make the difference between winning and losing. Team Giant-Alpecin has been focusing on this part of racing by organizing training sessions with downhill specialist Oscar Saiz, and the team aims to continue to search for improvement in every detail.” - TU Delft Sports Engineering Institute [4]

A Sports Engineering graduate student from the TU Delft has created a set of sensors that can be equipped on a road bike and trace the rider’s movement and behaviour on the bike. Together with research partner Team Giant Alpecin several test rides have been conducted in La Plagne, France, where riders were asked to make their fastest descent. The goal was to find the difference between riders and improve the performance of both the good and bad
descenders. The first analysis they did already resulted in some interesting outcomes and insights.

However, the amount of insight that can be obtained is limited and in this project we aim at creating a visual analysis tool for facilitating answering the question “why are some riders better than others?”

“Speed skating is a fun, but also difficult activity to perform. Balancing on thin blades, cross over in the corners and pushing sideways on a moving contact point is a challenge to many people. For world class speed skaters who are floating over the ice with speeds up to 60km/h, mastering the technique is essential.” - Skate & Science [10]

The best technique is, however, not the same for every speed skater. It depends on the body composition of the skater itself, the quality of the ice, and other factors like the air pressure. For a skater to be at their best, research has to be done on the optimal technique for each skater, and whether he or she has mastered this optimal technique. Winning or losing is a matter of milliseconds, so each stroke must be perfect. The Goal of this research done at the TU Delft BioMechanical Engineering department is to improve the individual skating technique and to give real time feedback to the skater on this technique. In particular, the researchers are developing an instrumented skate and aiming for real time feedback using smartglasses. Visual analysis can help this project by delivering a tool to inspect the data and answer the question “what is the most efficient way of skating in terms of technique and performance?” This will help identify what parameters should be feedback to the skater, such that they can be interpreted.

“Dutch baseball is internationally highly regarded, but according to the Royal Dutch Baseball and Softball Association (KNBSB) it is possible to reach an even higher level. Because the quality of the pitcher is critical to the performance of the whole team, the guidance of young pitchers to physical and technical top speeds comparable to American peers is very important. Therefore they should be strong enough to learn the proper technique and remain injury free, which in turn requires the right technique. The acquisition of this technique is a process of years supported in practice by strength training and mobility training. All these courses are mainly carried out based on experience and intuition: a lot of background knowledge is lacking. By working on three factors, the pitching level in the Netherlands needs to be raised.” - Project FASTBALL [2]

The first factor is the screening of youth selection players: the measurement of the physical development and the changes in strength and technique, in combination with the development of the throwing technique. Its main focus is on the development of the ejection shoulder and the likelihood of shoulder injuries. The second is building a training system that can calculate which muscles are used by a pitcher in “near real time”, how much muscle load is on them and what the outcome of the throw will be. This can be trained in the avoidance of damaging techniques and the achievement of the maximum speed. The third factor is the comparison of the effect of new ways of learning with the now conventional training methods. It is expected that the results can also be used in other overhand-sports.

Visual analysis of the pitch datasets is aimed on answering the question “what is the most ergonomic and efficient way of pitching in baseball?”

To summarize for all these three sports, we can say there is measurement data available which have several properties in common. Researchers all have the same problem: they miss a proper analysis tool that can help extracting the information they want, and visualize it in a comprehensible way. Often the analyst would like to have an exploratory view of the data since they want to be able to define new hypotheses. Therefore a good option is to use visual analytics. Visual analytics is a way of analysis that “integrates scientific disciplines to improve the division of labor between human and machine” [5], so that people can interactively
explore a comprehensible model of the data. The different sports have similar goals and data in an abstract level, therefore we develop a multifunctional visual analysis tool tailored to some specific sports, but generic enough to provide help for many more.

This tool would take previously gathered data from the different sports, and visualizes this in a comprehensible way, in order to extract the right information for the researchers. The data is not straightforward to visualize, given its heterogeneous and multi-dimensional nature: there are force vectors, positions, time-varying scalar data and several more data types. To be able to visually identify patterns in the data for the different datasets, a dimensionality reduction method is used, which provides a two dimensional view preserving the local neighbourhood.

We abstracted from the types of data and generated visualizations for each type. By making a lot of visualization parameters customizable and the visualizations interactive, the tool can be used for more sports than only the three mentioned before.

In this report, we will define the requirements for the different applications and abstract to common requirements. After this, we will see how the requirements can be translated to visual encodings and design. Here the main choices and difficulties will be discussed as well. Following, we evaluate the tool: a qualitative evaluation has been done with the researchers. This is followed by a conclusion, where we will see if visual analytics can indeed help in sports data research, and there is a proposal for future work.
In order to perform the task analysis, we first have to describe the data we have obtained from the different stakeholders, and make an abstraction and classification of this data. In the last section of this chapter we define the tasks and requirements from the stakeholders: the researchers from the three different sports projects, and we also define the common tasks to be achieved.

2.1. Data description
In this section, we describe in detail what the different datasets consist of, supported by descriptions of how these measurements have been obtained.

Cycling The cycling datasets each contain position and sensor data from a single descend down a mountain road. The sensors are attached to a road bike, and they measure GPS position, brake forces and steer and roll angles. The exact setup of the bike can be seen in figure 2.1, and the full list of variables can be found in appendix C.

During a measurement, a rider rode down the mountain, while staying in his own lane, and he was not allowed to pedal. The total time of the descend was measured. The datasets are sampled on the distance they have traveled from the start (by taking the nearest point on the ideal line). The total distance of the track was 1042 meter, so there are 1043 frames in each dataset, all representing one meter. This makes the descend with the smallest time the best performance.

There are currently 14 datasets for six different riders from Team Giant-Alpecin. Because the sensors were malfunctioning on the bad quality roads in the French Alps, the bike needs to be improved before more measurements can be taken.

Speed Skating The speed skating datasets each contain position and force data of a skater skating a straight part of the track. The measurements are done partly by placing markers on several important parts of the body, as can be seen in figure 2.2, and capturing the positions of these markers by high speed cameras. The other part of the measurements is done by a custom made instrumented skate, that can measure the forces of the skater in lateral and normal direction, and the angles of the skate with the track. This is done by gathering accelerometer data. From the normal and lateral forces of the skater we can also compute the global resulting force, which is the combination of these forces that makes the skater move. The full list of variables can be found in appendix C.

The length of the basic datasets depends on the time it takes the skater to skate across the straight part of the track, whereas the stroke datasets (derived datasets that each hold a
Data and task analysis

(a) Parameters to be measured

Figure 2.1: With the sensors from 2.1b all parameters in 2.1a can be measured. Figures taken from Lommers [7].

single stroke) are sampled to 150 frames each, making comparison easier. The speed of a skater is derived by the displacement of the central marker, placed on the sacrum. The stroke with the fastest average speed is the best performance.

There are currently 28 full datasets for four different speed skaters, of which three skaters also have a combined total of 30 datasets without the instrumented skates for comparison. From the 28 full datasets 86 strokes have been extracted. For some speed skating datasets not all markers were visible at each time frame, this causes missing data that has to be accounted for.

(b) Sensors on the measurement bike

Baseball The baseball datasets each contain the movement data of a single pitch (throw). The measurements are done by several inertial measurement units (IMU’s, a VICON motion capture system to be exact), placed on different points on the body. These measurement units measure the acceleration in three directions. Placing it on strategically chosen body parts allows you to calculate different body angles and movement speeds. The setup of the measurement units on the body can be seen in figure 2.3. In the datasets there are linear velocities and linear accelerations of several body segments (e.g. upper arm, measured at the center of mass); joint angles of several joints; angular velocities (rotational speed of a joint
2.1. Data description

(a) Upper body markers  (b) Lower body markers front  (c) Lower body markers back

Figure 2.2: The entire leg movement of the skaters can be reproduced by tracing these markers. Figures taken from van der Kruk [12].

(a) Measurement units on the body  (b) Reconstruction of the markers

Figure 2.3: The entire arm movement of the pitchers can be reproduced by tracing these markers. Figures taken from Project FASTBALL [2].

around an axis) of several joints; and joint moments (torque; product of force and distance from joint origin) of several joints. The velocities are interesting because they can roughly indicate the ball speed of the pitch. The joint moment is interesting because this is a measure of how much load is applied to the joint: the bigger the load, the bigger the risk of an injury. The full list of variables can be found in appendix C.

There are 121 frames in the dataset, where each frame represents a millisecond of the throw. There are several events saved in the dataset, where the value of the event is the frame in which this event happens. Each dataset is centered around the ball release event (this event happens at frame 61). The pitch with the highest ball velocity is the best performance.

The pitchers in the datasets are members of the Dutch national selections. There are 11
pitches from the A team, and 21 pitches from the AAA team, which is the youth team. Together this makes for 32 throws of 13 different pitchers.

2.2. Data abstraction
All variables in the different datasets can be classified according to the type of data. This will guide the visual encoding. Below is a list of different data types and the variables that are of those types. In Nieuwenhuizen [8] you can find a summary of data types and the different kinds of visualizations that can be made for each type of data. We can identify four main types of data, that all have a temporal aspect.

- Scalar data: time series of scalars (numbers)
  - Force magnitudes (speed skating)
  - Angles (in the speed skating data, yaw, pitch and roll angles from the instrumented skate are separated)
  - Speed (speed skating)
  - All relevant cycling variables except for position

- Vector data (forces, velocities, accelerations, moments): time series of vectors, mainly in three dimensions.
  - Global force (speed skating)
  - Velocities and accelerations (baseball)
  - Accelerometer data (speed skating)
  - Joint moments (baseball)

- Angle data: time series of multi-dimensional angles (e.g. Euler angles) expressed as vectors.
  - Joint angles (baseball)

- Positional data (marker positions, GPS): time series of multi-dimensional positions.
  - Marker positions (speed skating)
  - GPS position (cycling)

There is a special category of data, that can describe something about the participant or about the dataset performance. These data are not time series. Examples are events in the baseball datasets, which indicate the timing of certain events in the dataset. Another example is the maximum exerted force by a speed skater.

2.3. Task analysis
The initial requirements for the visualization tool have been gathered during a multi-cycle process of several stakeholder interviews. These interviews were held with the people who gather and analyze the data for baseball, speed skating and cycling. From the interviews several tasks and requirements came forward, which are listed below. By implementing parts of the design and thus prototyping, more specific tasks came forward, which helped in improving the design further.
2.3. Task analysis

**Cycling**  The goals for cycling consist of the following tasks:

- **Main task**: Finding the differences in behaviour between good and bad descenders

- **C1.** Identify why some persons are simply better than others (*the task abstraction is to: compare individuals using scalars* - later referred to as *T1*)

- **C2.** Inspect extra parameters, such as pedaling power, when they are added to future datasets (*inspect scalar data* - *T2*)

- **C3.** Identify and compare groups that behave similarly (identify groups - *T3*, compare groups using scalars - *T1*)

- **C4.** Select and inspect only a part of the complete course, for example a single corner, as some riders make a significant difference with others in a particular corner. How they differ here might explain the difference in performance (*selection of time periods* - *T4*, compare individuals using scalars and positions - *T1*)

- **C5.** Compare leftward and rightward corners of one person with the average leftward and rightward corners. The difference in time difference would show if a rider is relatively better at a specific side (compare groups using scalars - *T1*)

**Speed skating**  The goals for speed skating consist of the following tasks:

- **Main task**: Finding the most efficient way of skating in terms of technique and performance

- **S1.** Identify a relationship between different angles and forces and the speed of the skater (*compare groups and individuals using scalars and vectors* - *T1*)

- **S2.** Inspecting an image of the skater with the force vectors projected on it (*inspect/discover trends using vectors and marker positions* - *T2*)

- **S3.** Detect the strokes in a skating dataset: inspect speed, width, forces of each stroke and compare these stroke datasets to each other (*select data by stroke* - *T4*, compare individuals using scalars - *T1*)

- **S4.** Inspecting if there is any difference between the movements of skaters with or without the instrumented skates (*compare groups and individuals using vectors and marker positions* - *T1*)

**Baseball**  The goals for baseball consist of the following tasks:

- **Main task**: Finding the most ergonomic and efficient way of pitching

- **B1.** Finding a correlation between timing and magnitude of applied forces and resulting ball speed (*compare groups and individuals using scalars, vectors and angles* - *T1*)

- **B2.** Identify whether the sideways motion of the arm displays a smaller torque on the medial elbow side on different throw styles (*compare groups and individuals using vectors, angles and marker positions* - *T1*)
From these tasks some core tasks could be abstracted. Some of the tasks overlapped between the different sports. The list of abstracted tasks is given below.

**T1.** Compare individuals and groups using scalars, vectors, angles and positions (marker and GPS)

**T2.** Inspect and analyze (discover trends) the values of different types of measurements

**T3.** Identify groups of datasets based on given measurements

**T4.** Selection of time periods from data

In the chapter about the visualization design (chapter 3) you will find how these tasks are fulfilled.
In this chapter we will present the design of our visual analysis tool based on the requirements, tasks and the given data. The chapter is ordered following the list of abstracted tasks. The first section is about visualizations fulfilling tasks $T_1$ and $T_2$: inspecting, analyzing and comparing individuals and groups using scalars, vectors, angles and positions. The second section presents how groups of similar datasets are identified ($T_3$). The third section is about selection of time periods ($T_4$).

3.1. Visualizations

This section describes how main tasks $T_1$ and $T_2$ are fulfilled, which are about inspecting, analyzing and comparing different kinds of data. This can all be done using visualizations, as long as the data is presented in a suitable way.

In order to visually compare different datasets, it is important to identify individual representations. Each dataset has its own color, which is maintained in every visualization in the design. This color is given based on the participant the dataset belongs to. Each participant has its own hue, and each dataset has a different tone of that hue, given that each participant can have multiple datasets. We are aware that using a different tone of a certain hue indicates an order in the datasets, but given the fact that we want to reserve one hue for each participant to allow for separation between participants, and that we want to separate different datasets for each participant, it is the only option left. Color coding is already a visualization of a scalar variable. This way, it is easy to spot datasets that belong to the same person, and to compare different datasets.

To have the ability to inspect and analyze the values of different types of measurements, different kinds of plots are needed. Looking back at section 2.2, we see we have four different classifications of data: scalar data, vector data, angle data, and position data. For the visualizations, it is important that the users can easily interpret the data. Therefore, different plots have been designed for the different kinds of data, each addressing one or more data types. For each plot we will describe in the following subsections how it can be used to inspect, analyze and compare datasets.

Serving different end-users with different interests and preferences is easiest when you allow them to customize the way they see the data to their own desires. Therefore, for most plots, it is possible to choose what data you want to see, and how you want to see it.

The last subsection explains how all these plots are linked together to enhance the user experience, and to make visual analytics of different kinds of variables possible, thus to fulfill the tasks.
3.1.1. Line plot
The line plot (figure 3.1) is a common visualization for scalar temporal data. The line plot is the most basic plot in the tool. Since ‘time’ is on the horizontal axis, it functions as a time line. By default, the ‘time’-variable is a frame in the dataset. For the cycling sport, it is also possible to have the actual time in seconds as time variable at the horizontal axis.

On the vertical axis this plot can show all scalar temporal data, and the magnitude of the vector data. Multiple variables and multiple datasets can be selected at the same time. The visible variables are shown in a list, and upon hovering over the variable name, all lines for that variable will be highlighted. This gives you the opportunity to inspect multiple variables in the same plot without losing focus of one variable in the clutter of lines.

When multiple variables are selected, it is possible that the scales of these variables are completely different. To be able to compare these variables, there is an option to ‘normalize’ the values. All variables will then be mapped to a range between 0 and 1, allowing the user to compare variables with different magnitudes and find relations between variables without absolute value comparison.

Figure 3.1: Line plot. Four left strokes of different speed skaters are compared by left and right normal force. The left normal force is highlighted so that these lines can be identified between the others.

3.1.2. Vector plot
To visualize vector data, we design what we call a vector plot (figure 3.2). For certain vector variables showing the components of the vectors on the line plot would work, for example when you only want to inspect the force in a certain direction. Since this is not the case for all vector variables, we have to design a visualization that is able to show the change of a vector in time, which is not possible on a line plot. The vector plot is a visualization of a unit sphere, with a unit vector originating from the center of the sphere. Unit vectors are used to make it easier to compare the orientation of different vectors to each other, since you have a smaller change of missing vectors because their magnitude is very small compared to others. This plot is used to visualize vector data, and we have specifically chosen for the direction of the vector. The magnitude of the vector can also be important, so this is shown in the line
plot, since this can be considered a scalar temporal variable. The magnitude is shown for the vectors where inspecting components would not work.

Because a line of the trace of the vector would be completely cluttered, we have chosen for an animation to display the change of the vector direction in time. Since the datasets already claimed the color dimension, and since it is convenient to stick to a unit vector in the sphere (to keep the tip of the vector at the edge of the sphere), the only dimension left for visualizing the magnitude of the vector is the thickness of the vector. Since multiple vectors can be plotted at once, and they are likely to point in roughly the same direction, this would not be easy to evaluate. Therefore, the magnitude has been moved to the line plot, which brings the advantage that it can be compared to other magnitude values and scalars there.

A ghost trace (previous directions of the vector) can be visualized at will. This ghost trace is a transparent representation of the vector in the previous time steps. Visualizing this allows you to analyze the vector over multiple time steps at once.

When you are visualizing a variable that is related to a joint (e.g. wrist, elbow or shoulder in baseball), a representation of a joint is created in the sphere, to be able to relate the direction of the vector to the real world.

![Figure 3.2: Vector plot. The unit vectors show the direction of the variables. The speed of the hand (in the left sphere) is directed in a similar direction. In the moment sphere (on the right) you see the main direction of the moment in the wrist, which differs a lot for the four different datasets. This can be an indication of a different technique. Ghost traces are shown 5 steps back.](image)

### 3.1.3. Movement plot
To visualize the angle data and the marker position data, the movement plot (figure 3.3) was designed. Since the baseball and speed skating datasets contained data on how the body moves, and deducing the movement posture of the body based on the individual variables is not possible, it made sense to show a visualization of a moving body. Since the cycling datasets have steer and roll angles, there is also an visualization of a bicycle moving as described in the data.

The posture changes on time, and the movement can be stepped forward and backward, and the user can zoom in on and turn the camera to a particular part of the model so that each movement can be analyzed thoroughly.

For both the body [9] and the bicycle [11] a freely available mesh has been used. This mesh
could be manipulated to move according to the data. The data needed to be manipulated in a way that it was applicable to the way the mesh (in particular the skeleton of the mesh) was defined. For the skating data this meant that from the positions of the markers a global coordinate system could be created. Using this coordinate system the rotation matrices of the joints can be calculated, and these can be applied on each animation step to the respective bones, relative to the parent bone. For the baseball data, where the data was defined in Euler angles, the coordinate systems in which the data was measured were different from the coordinate systems of the mesh. A mapping had to be created that made the transformation from one coordinate system to another. A detailed description of these transformations can be found in appendix D.

As an addition to the angle and marker position data, there is also some vector data in the movement plot. Examples are the force vectors from the speed skating datasets and the brake forces from the cycling datasets. These are added to inspect trends in the position and angle variables compared to the vector variables.

The movement plot allows you to hide or show different parts of the mesh to allow for focus at a certain movement, or otherwise to inspect more information at once. Examples of parts that can be hidden or shown are the force vectors or the skeleton. The movement plot also allows you to customize the way the mesh looks in terms of opacity and wireframe, to optimize the visibility of overlayed meshes. Here you can also choose the length of the ghost trace of the animation, allowing you to see how the movement progresses in a still image.

(a) Three speed skaters are compared, and the resultant force vectors are plotted. The meshes are made slightly transparent to be able to see the contours of all skaters.

(b) Three bicycles are plotted with the brake force vectors. You can see the roll angle of the bicycles in the plot is different, which indicates a different approach of the corner. You can also see that the green cyclist is not braking at this moment.

(c) Three pitchers are reconstructed in this image, performing their throw. You can spot the difference in their throwing styles. The orange pitcher has his arm more extended to the back.

Figure 3.3: Movement plot.
3.1.4. Trajectory plot
To able to inspect and analyze the remaining positional data, the trajectory plot was designed. The trajectory plot appears in two different forms. Since the positional data in the cycling datasets and in the speed skating datasets are of a different kind, they need different visualizations.

Task C4 states that it should be possible to select a part of the course, specifically corners. In order to do so, one should have an overview of the course on a map. Since there are GPS positions in the dataset, this could easily be done with Google Maps [3].

On the map (figure 3.4) there is a line following the course of each rider in the respective color. When the animation is running, there will also be a representation of the rider on the course, that will be marked when this rider is braking. There is a possibility to move the start and end point of the course, which are indicated by a start and finish flag on the map. When changed, the animation will only run from the start to the end of the selection, and all plots will be updated accordingly. When time is used as time variable instead of the distance, the animation will give a playback of the descends of all riders against each other.

According to task S3 strokes have to be detected from the skating datasets. In order to verify these strokes and see how they align a plot is designed that shows the course of the skater during each stroke by using the marker positions on the lower back of the skater (figure 3.5). This way you can see which strokes follow up on each other and how the strokes differ in width and length, which can be a measure of efficiency.

3.1.5. Event plot
The scalar data that does not consist of time series is embedded in the program somehow. One group of this data was worth designing a separate plot. Task B1 speaks of finding a correlation between the timing of applied forces to the resulting ball speed. Since there are timings based on forces saved in the events data, these events are relevant for this task. Plotting the events in the line plot cluttered the data too much, so a separate plot was made for the events (figure 3.6). Since all events are centered in a smaller time frame than the entire dataset, a smaller scale could be used for this plot. The events are all plotted as dots on a time line, and clicking one of these dots makes the animation go to the respective time frame.

3.1.6. Linked views
To facilitate a data analysis using all possible plots (and thus all variable types) that are relevant for the selected datasets, all these plots are somehow connected to each other. This allows us to fulfill tasks T1 and T2, inspecting, analyzing and comparing different kinds of data. Dataset selection is synchronized over all plots. The animation frame, which is the time step that is visualized in the animation, is visible in all other plots. The line plot and event plot have a slider going over the time axis, and the vector plot, map plot and position plot each have their own animation. Selecting speed skating force variables to be shown in the vector and line plot will add a vector visualization in the movement plot, as mentioned before. The magnitude of a vector visualization will also be shown in the line plot, to be able to compare the direction and the magnitude of a vector at the same time. In multiple plots we allow the filtering of data (section 3.3). The start and end time of a dataset can be adjusted, and this will be represented in the line plot by a smaller domain on the time axis, as well as by a movement of the start and end point on the map plot.
3. Visualization design

Figure 3.4: Trajectory plot for cycling. By sampling the dataset on time, we can see that the blue rider is faster than all others. The orange rider is braking here, as shown by the extra circle. This is probably the reason this rider is the slowest.

Figure 3.5: Trajectory plot for speed skating. Three strokes are shown here. The traces of the upper two strokes are quite similar. The other stroke is in the other direction.
3.1. Visualizations

Figure 3.6: Event plot. Outliers and slight differences between timing in the pitches are very clearly visible.
3.2. Identifying groups of datasets

To determine which datasets form a group, based on certain parameters (task T3), one should visually compare different datasets and determine which are alike. Humans will be able to identify relations between one or two variables for multiple datasets, but for a larger number of variables it is an impossible task. Therefore, a dimensionality reduction method is applied to be able to explore this high-dimensional space in a two-dimensional representation. The result is an embedding, a two-dimensional projection where each point represents a dataset. This embedding can be used to identify groups.

A visual example of an embedding is given in figure 3.7. When you see a higher density of dots, it is an indication of a group. Once you have identified a group of similar datasets, you can save it as a group. In order to be able to analyze the group and the difference with other groups, we need to summarize the data. A group is generally represented by the average of all the datasets that form the group. We define the average of a group of datasets as the numerical average for each time step for each variable in the dataset. In this case, each component of a vector is considered a separate variable. This deserves more attention because the numerical average of components is not suitable for averaging angles, for example. Because the average is calculated for each time step, it is important that all datasets have the same amount of frames. When calculating the average, the standard deviation of each variable is also calculated, as well as the standard deviation of the magnitude of vector variables.

The embedding is visualized in the so-called embedding plot (see section 3.2.3). This plot can for example be used in the following way. Once you have identified a group of similar datasets, and it turns out that all these datasets have a good performance (or a bad one), one can inspect the variables in the dataset and find out why these datasets perform better (or worse) than others. This can help to form hypotheses on what variables can make a winner.

![Figure 3.7: The datasets in the red selection can be considered a fast group. The map seems to indicate that there are differences in the data between well performing and not so well performing datasets.](image)

The embedding should preserve the structure of the original data as good as possible. In our case, we are interested in the grouping. Therefore we use t-SNE.

3.2.1. t-SNE embeddings

As mentioned earlier, in order to make it possible to visually compare the multivariate datasets to each other, we apply dimensionality reduction. After some research on dimensionality reduction methods, t-distributed stochastic neighbour embedding, or t-SNE [13] came out on top. This was selected because in contrast to many other dimensionality reduction methods, t-SNE tries to keep similar data points together, instead of keeping dissimilar data points far apart. t-SNE performs dimensionality reduction to two or three dimensions, ideal for visualization in a scatter plot. t-SNE simply does the following: If you have a collection of datasets
3.2. Identifying groups of datasets

\[ X \text{ in the feature space, a mapping } Y \text{ in the projection space is created,} \]

\[ X = \{x_1, \ldots, x_n\} \xrightarrow{t-SNE} Y = \{y_1, \ldots, y_n\} \]

that defines coordinates for each dataset, representing their place in the two dimensional space created. This space is defined by modeling each dataset as a point, while preserving the local neighbourhood, which makes it an embedding. Therefore, datasets that appear close to each other in the embedding, are datasets that are similar based on the defined distance in the high dimensional space. The definition of neighbourhood depends on the definition of distance. t-SNE does not preserve absolute distances to points. The distance between points outside the neighbourhood is not preserved.

3.2.2. t-SNE algorithm

As input value, t-SNE needs the distance between each pair of datasets \((x_i, x_j)\), therefore, say each dataset \(x_i\) has a collection of \(K\) curves \(c\) (measurements), and each curve has an entry for each of the \(T\) time steps. These entries can be a scalar, position or vector. Since these different measurement types have to be treated differently, and according to their meaning in the real world, defining a good distance measure is difficult. Between scalars you have for example the difference between a measurement in meters and in kilometers, which is an easy mistake to make, but makes for large differences in the distance measure. Or try comparing a doubling of a distance to a doubling in temperature, for example, they do not mean the same. The same problem exists for differences between vectors, and comparing between scalars and vectors might prove even more difficult.

We are comparing apples and pears, so we have to find a way to be able to compare them. If you do not do this, one variable with very high values could dominate the distance measure, and then just this variable would define the difference. We opt for standardizing the data using Z-Score normalization \([6]\), by calculating the mean \(\mu\) and variance \(\sigma\) of each variable over all datasets, and replacing each value \(x\) by its z-score \(z\), using the following formula:

\[ z = \frac{x - \mu}{\sigma} \]

The result is a collection of datasets \(x_i\) with \(K\) curves each. Each curve \(c_l\) has a normalized entry for each of the \(T\) time steps. This entry is the amount of standard deviations the original value was away from the mean value.

\[ x_i = \{c_1, \ldots, c_K\} \quad c_l = \{c_{l_1}, \ldots, c_{l_T}\} \]

For the distance calculation, the datasets are transformed to simple number matrices, of size \(4K \times T\), containing 4 columns for each variable, representing three dimensions of the vector and the magnitude of the vector. The distance for a vector is the sum of the distance of the magnitude (which is considered a scalar) and the vector components. For scalar variables, we only define the fourth column, and for position variables we amend the fourth column, since these are not meaningful for these types of variables. Angle variables are treated as vectors. The matrices have a row for each time step.

Then the distance between two datasets \(i\) and \(j\) can be defined by the following formula:

\[ \delta(x_i, x_j) = \sum_{l=1}^{K} \alpha_l \times f(c_l^i, c_l^j) \]

where \(\alpha_l\) is a defined weight of curve \(l\), and distance function \(f\), that defines the distance between two curves, is defined based on the type of measurement. Since different variable types cannot be treated equally, \(f\) needs to be different for different variable types.
3. Visualization design

\[ f(c_i, c'_i) = \begin{cases} 
  s(c_i, c'_i), & \text{if } c_i \text{ and } c'_i \text{ are scalar curves} \\
  v(c_i, c'_i), & \text{if } c_i \text{ and } c'_i \text{ are vector curves} \\
  p(c_i, c'_i), & \text{if } c_i \text{ and } c'_i \text{ are position curves} 
\end{cases} \]

\( s \) is a function that calculates the distance between two \( 1 \times T \) vectors, defining a scalar variable or the magnitude of a vector. This is done using the euclidean distance between the two scalars.

\[ s(c_i, c'_i) = \sum_{t=1}^{T} (c_{i,t} - c'_{i,t})^2 \]
\[ s(c_i, c'_i) : 1 \times T \to 1 \]

\( v \) is a function that calculates the distance between two \( 3 \times T \) vectors, where each column defines a dimension of a vector or any other three dimensional variable. This is done using the dot product of the two vectors.

\[ v(c_i, c'_i) = \sum_{t=1}^{T} (1 - <c_{i,t}, c'_{i,t}>)^2 \]
\[ v(c_i, c'_i) : 3 \times T \to 1 \]

\( p \) is a function that calculates the distance between two \( 3 \times T \) position vectors, where each column defines a dimension of the position vector. This is done using the euclidean distance between the two vectors.

\[ p(c_i, c'_i) = \sum_{t=1}^{T} \|c_{i,t} - c'_{i,t}\| \]
\[ p(c_i, c'_i) : 3 \times T \to 1 \]

We are aware that this is just one possibility, other options could be explored to improve the grouping of datasets.

3.2.3. Embedding plot

The embeddings that result from the previously described method are plotted in the embedding plot (figure 3.8). This plot is a dot plot of the embedding positions, where each dot represents a dataset. The absolute position of a dot in the plot has no special meaning, coordinates do not have a concrete meaning. The only thing that has meaning in this plot is the proximity of one dot to another. When two or more dots are close, this means that the datasets are similar, based on the given distance. The embedding plot (and the embedding algorithm) fulfills the main task on the ability to identify groups of similar datasets. The definition of similar in such a heterogeneous space is difficult, therefore we provide the user with the criteria for the distance, that can be reviewed from the legend of the embedding plot. Here you can also see the legend for the colorings of the dots. The dots can be colored according to another variable, for example according to participant, which allows you to see which participants behave alike, or according to performance, which allows you to identify groups with similar characteristics that perform similarly. Selected datasets will be marked by a ring in the color of that particular dataset. Datasets that are marked as a group will also be marked by a ring in the color of the group. The process of creating the embeddings is customizable, because you can choose which datasets to incorporate, and how much each variable will be taken into account. The plot is interactive, meaning that you can select and unselect datasets from it, and save these datasets as groups for comparison using the previously described plots. How this is done is described in the next section.
3.2. Identifying groups of datasets

Figure 3.8: Embedding plot. An embedding is shown colored according to participant (left) and according to performance (right). This way you can easily see which participants perform best, and which groups of datasets represent good performances. The points can be selected for further analysis and to identify differences.

3.2.4. Visualization of groups

In addition to comparing the individual datasets, task T1 also included the comparison of groups of datasets to each other based on different measurements. Generally, groups are represented by the average of all the datasets that form the group. As mentioned earlier, this average is defined as the numerical average for each time step for each variable in the dataset.

Groups are also encoded in color. Since each dataset has a performance measure (generally 'speed'), the groups can be arranged by the average (also a numerical unweighted average) performance measure of the datasets within the group. The best group obtains the brightest color, and the subsequent groups all get some tones darker. This way, it is always clear which is the best performing group. Color coding is already a visualization of a scalar. In this case, it is used to identify the best performance.

For the visualization of groups of datasets in the line plot, the average line for the selected variables is plotted. The average line is accompanied by a range that indicates the 95% confidence interval by taking a distance of two times the standard deviation from the average line. It is possible to change the plot to a 'group'-view, so that it only shows the groups. The best group (the group with the best average performance of the contained datasets) is shown as a baseline, and every other group is visualized by a line showing the difference to the best group for each variable. See figure 3.9 for an example of a group in the line plot.

For the visualization of groups of datasets in the vector plot, the average vector for the selected variable is plotted. There will be a circle at the tip of the average vector, that is scaled by the standard deviation of the respective variable in that group. This means that when there is a big circle on the tip of a vector, the vectors of the individual datasets in the group are pointing in different directions. The vector plot allows you to choose whether you want to see the variance or not. See figure 3.10 for an example of a group in the movement plot.

For the visualization of groups of datasets in the movement plot, the visualization of the most representative dataset is shown. The most representative dataset is used because the average of multiple positions or angles is not necessarily meaningful in the posture animation. The most representative dataset is selected by taking the dataset having the smallest difference with all other datasets. This difference of one dataset to another is defined by taking the sum of the squared difference of each variable in each time step. Animations of the remaining datasets are shown as ghosts in the background. Ghosts are the same visualizations as the ones for the individual datasets, but they are transparent. This is done because when
Figure 3.9: Line plot. Comparing two groups of two strokes of different skaters. The lateral force is currently highlighted, and this shows a higher average of the yellow group at the end of the stroke, but a greater variance in the orange group. The yellow skater thus has a more stable technique.

you show them normally, the view of the group animations is cluttered because of the animations of all containing datasets. Making them transparent lets the group animations stand out, but still preserves the information of what datasets make up a group. See figure 3.11 for an example of a group in the movement plot.

For the visualization of groups of datasets in the trajectory and event plot, the average performance is shown in the same way as an individual dataset.
3.2. Identifying groups of datasets

Figure 3.10: Vector plot. The unit vectors show the direction of the variables. In the moment sphere (on the right) you see the main direction of the moment, which differs a lot for the two different datasets. The yellow vector is the average of the orange ones.

Figure 3.11: Movement plot. The most representative dataset of the yellow group is shown, along with a ghost that represents the other dataset from the group. The pink skater is a stroke in the opposite direction that is not in the group.
3.3. Selection of time periods

There are two tasks that speak of selecting time periods from the data. Both have a different solution.

In order to fulfill task S3: ‘detecting strokes in the dataset’, the original speed skating datasets were cut at the frames where the normal force went to 0. A left stroke was detected when there is force applied to the left skate, and a right stroke was detected when there is force applied to the right skate. Therefore, the strokes may (and actually should) overlap.

In order to fulfill task C4: ‘select and inspect only a part of the complete course’, there is an option to choose the start and end time of the plotting of the dataset. This smaller range will be shown in the line plot, and the animation will only play within these boundaries as well. Having an option to calculate embeddings based on a smaller time frame is future work.

3.4. Conclusion

We will conclude by summarizing how the proposed design is able to fulfill the tasks that emerged from the task analysis (chapter 2).

**Cycling** For cycling, most tasks are only doable when new measurements are done, because the amount of datasets is insufficient to discover trends, and the data is not complete nor accurate for all datasets, due to malfunctioning sensors. The measurement bike should be updated for new data, which is outside the scope of this project. Therefore, the visualization tool is created such that it can accommodate the new data, but the more specific visualizations are not yet created. We evaluate what can be done with the current data.

Task C1, identifying why some persons are better than others, can be done by grouping all datasets of a person together and comparing this group to other persons groups using the line plot and possibly the GPS plot. This way you can also identify groups of persons behaving similarly (task C3). This can also be done using an embedding.

Task C2, inspect extra parameters that can be added in future datasets, is something that can be done, as long as the variables are of the same type as the variables that are currently supported in the tool.

Task C4, selecting only a certain part of the course can be done using the line plot menu. You can change the domain of the X axis, this allows you to select a certain part of the course. Use the GPS plot to see where the corners are located.

Task C5, doing an analysis comparing different corners in the data is currently a bit cumbersome. You could do this analysis manually, by comparing parts of datasets to each other, but not automatically one corner against another. This would require a completely different approach, and therefore it is not implemented.

**Speed skating** For speed skating, the tasks can be fulfilled, but the visualizations can be improved to extract more information out of the data, in a more intuitive way.

Task S1, identifying a relationship between different angles and forces and the speed of the skater, is something that can be done using an embedding. Since the performance measure of a speed skating dataset is the forward speed, when you color the embedding according to performance you will see how fast skaters and slower skaters are related. If you create clusters for both groups, you can compare them. Using the line plot you can compare the forces and the roll, pitch and yaw angle components.

Task S2, inspecting force vectors on an image of the skater, is fulfilled by looking at the animation, where you see the lateral, normal, and resulting forces projected on the three-dimensional mesh.
3.4. Conclusion

Task S3 is fulfilled by the datasets in the 'Speed Skating - Strokes' folder. These are extracted from the original datasets by checking when the force on one of the skates was equal to zero. The information of the strokes is displayed in a legend tooltip on the line plot.

Task S4 can be fulfilled by finding datasets of the same person in the 'Speed Skating' folder and the 'Speed Skating - Normal' folder. It is important that the datasets are somehow aligned, so you can see the movements in superposition. To make this easier the option 'Move mesh' could be unchecked, this makes the upper bodies aligned, and only lets the legs move.

Baseball For baseball, the tasks can be fulfilled, but the visualizations can be improved to extract more information out of the data, in a more intuitive way.

Task B1, finding a correlation between timing and magnitude of applied forces and resulting ball speed, can be done using an embedding. Since the performance measure of a baseball dataset is the resulting ball speed, when you color the embedding according to performance you will see how fast pitches and slower pitches are related. If you create clusters for both groups, you can compare them. Using the line plot you can compare the magnitude and timing of different torques.

Task B2, identifying and comparing different throw styles on torques, is a bit more involved. To do this you should manually identify different throw styles, or you might create an embedding based purely on angles, and use that as a guide to see which throw styles are definitely not alike. Now use the vector and line plots to inspect the angle and magnitude of the torque for the different throw styles.
In order to evaluate our visual design and interaction for the exploration of the sports datasets, and to verify that it can show which measurements influence the performance the most, the design has to be tested.

In addition to several intermediate demos and test rounds, a qualitative evaluation has been conducted at the end of the project. The testing procedure is described in this chapter, along with its results.

4.1. Qualitative evaluation
The test subject were a minimum of one for each sport involved in this project. The experimental setting was as follows. After an initial introduction and demo on the possibilities of the tool, we allowed the users to interact with it themselves. They had to perform a series of tasks related to their sports. These tasks will be leading them towards the answers of the questions they have presented during the meetings. Afterwards, the users were allowed to freely explore their data. A questionnaire with open questions was then answered by the users. In addition to the usability questions, the test subjects were asked how useful each feature of the tool was to them, how well each test task resembled their real tasks, and whether the tool met their expectations and where it could be improved to better suit their needs. The complete forms can be found in appendix E.

In the following subsections we will go through the lists of tasks created for the testing for each sport, that should resemble tasks the researchers want to perform with the tool.

Cycling

- **Task 1:**
  - Create a group for each person
  - Select different variables and compare the riders using the line plot
  - Click the switch in the GPS plot to show the animation by time
  - Play the animation
  - Draw conclusions

- **Task 2:**
  - Create an embedding
– Create groups of fast and show riders
– Compare the groups using the line plot and the GPS plot
– Name the most relevant variable(s)
– In which corner does the fastest group make the biggest difference?
– Is this reflected in the time difference?
– Zoom in on this corner, and see what the difference is between the groups in the most relevant variable(s)

### Speed skating

- **Task 1:**
  – Compare two skaters based on one variable

- **Task 2:**
  – Inspect resultant forces in combination with the Center of Mass. Also try normalizing the forces by weight.

- **Task 3:**
  – Create embedding based on angles and forces
  – Create subembedding of the left strokes
  – Create appropriate groups
  – Select angle and force variables (one by one or all at the same time)
  – Inspect variables (using the line plot or animation)
  – Draw conclusions

- **Task 4:**
  – Find matching and working datasets from the 'Speed Skating' and 'Speed Skating - Normal' folders
  – Compare the animations (use markers)
  – Draw conclusions

- **Task 5:**
  – Compare two right strokes from two different skaters based on the sacrum-foot distance and the maximum left and right forces
  – Draw conclusions

### Baseball

- **Task 1:**
  – Create embedding based on moments
  – Create appropriate groups
  – Select moment variables (one by one or all three at the same time)
  – Inspect magnitudes (eventually display vectors as components)
4.2. Test results

The test sessions were done by following the testing procedure and letting the users explore the tool. While doing their tasks, they came up with improvements for future work, and they also discovered some bugs. All these things were noted and a list of the most interesting improvements will be given below, separated per sport. There is also a description of the reaction of the test subjects to the tool. In general, the reaction was positive, and all subjects would like to use the visualization tool (preferably with the mentioned improvements) in their data analysis. This confirms that the observed need for a data analysis tool is really there, and that visual analytics can help solving this need.

Cycling For cycling, the test was mostly about showing the potential of the tool when there is more data available. Currently, plans are being made to improve the measurement bike, so in the near future this should become reality.

This potential was seen by the test subject, and he was even considering if the tool could be used to analyze other data than the descending data, for example the data that is generated during every training session, or time series of data generated over a year for each cyclist in the team. In theory, this would all be possible. The usefulness of the map plot and the movement plot was difficult to evaluate, since there should be more data incorporated in these plots, and a zooming function in the map would greatly improve its usefulness as well.

A complete list of the future work is given below.

- Zooming on the map plot for more detail
- Filtering of the data
- Easy adding of new and other data and parameters

Speed skating For speed skating, two test rounds have been done, since the first test session was the first of all the tests, and usability bugs were found, that were quite essential to the functioning of the tool. The second test session was held after these bugs were fixed, and this time we could go more in depth about the features of the tool.

There were two test subjects for both sessions: one is the developer of the instrumented skate, and the other is a master student who will perform the kinematic comparison between separate skaters. Both subjects would like to use the tool for their data analysis, since they feel they can understand the data better and they can discover new relationships and insights on the data using the tool. They state the system can be improved concerning usability. Most features were found useful, but the vector plot consistently scored low. This can be explained by the fact that the force vectors are already added to the movement plot, so there is small added value of the vector plot. The value of the embeddings and clusters was not very clear to the subjects. One main problem that appeared during the testing was that the data was
unfiltered. This causes a wrong distance measure, which makes the outcome of the embeddings not very useful for the current data. Most tasks were found very relevant to the research. The person who did not understand the embeddings found task 3 (on embeddings) irrelevant. However, the other subject scored it 5 out of 5 for relevancy. Task 5 (comparing two skaters) was too specific for both their needs.

From both sessions some points for future work came forward, which are listed below.

- Filtering of the data (to a fitted line)
- Deal with missing data (e.g. missing markers)
- Deal with new data
- Have the ability to load a custom mesh (there are constraints to the meshes that can be used: it should for example be able to deform following the movement of the skeleton, and it should have a proper definition of the required joints)
- Apply own rotation matrices directly to the mesh
- For movement scientists one should hide the data errors, but for sports engineers one should make clear what data errors there are and what the program does to solve them

**Baseball** For baseball, there was one test subject that was particularly interested in the movement animation. Last minute before the test session an animation of the full body movement was created, using new datasets. This was to show the potential of the animation. The subject was enthusiastic about this, and really liked to use this in an improved form. For this to happen, the way the data is loaded in the system has to be changed, so that the researchers have more control over what data they see. This is a finding that came forward in all three tested sports.

The subject did find an interesting fact in the data that he had not spotted before, which already proved that exploring the data in a visual analytics way can help forming new hypotheses. In this case combining the event plot with the movement plot and the performance led to the discovery that some slower pitchers had early foot contact.

Apart from this, most visualizations were just not working well enough for the baseball analysis, since the direction in the vector plot could not be related to the joint yet, and the line plot was too cluttered by vector components to do any analysis. So for most features there are suggestions for future work. It was, however, clear that there is great potential in this visual analytics. The most interesting improvements that came forward were to move the distal end of the joint in the vector plot (see the joint in the right sphere in figure 3.2) like it does in the data, to really show the movement together with the force in an isolated surrounding. The second improvement was to plot the movement animation of each person at their respective timing of a certain event. An example would be to plot all pitchers at the moment of foot contact, to be able to compare their actions around that event, instead of comparing them at a certain time. A complete list of the suggested future work is given below.

- Have a customizable body segment length for the movement animation for each participant
- Show each component of a vector in a different way to improve separability
- Move the distal segment of the joint in the vector plot combined with the joint moment for an isolated view of the moment and the movement in a single joint
- Color the embedding on the minimum or maximum of each possible variable
4.2. Test results

- Color the animation based on performance measure
- Plot all movement animations at the timing of a certain event in that dataset
- Moving the camera in the movement animation to face a specific anatomical plane (sagittal, coronal, transverse)
Conclusion and future work

In this thesis we present a design for a visual analysis tool for sports research. The developed tool has separate visualizations for scalar data, vector data, angle data and position data. Using these visualizations, different sports datasets with time-varying data can be inspected and compared to each other. This tool can help researchers to define new hypotheses on their data. The tool has specific visualizations for three sports: cycling, speed skating and baseball, but can be used to analyze datasets for all different sports. Using a dimensionality reduction method, the user of the tool can get an overview of which datasets are similar to each other, based on defined parameters. Combining this method with the performance of a dataset, it is possible to find out what parameter values cause a good performance. This knowledge can help the researchers in improving the sports performance of the athletes.

The usability of the current tool is limited, however, the researchers saw great potential in using visual analytics for their research. They were convinced by the way they could explore their data without having to create their own plots for each part they wanted to inspect, and how all plots were updated in real time.

For future work the main challenge is the way the data is entered into the tool. This should be improved so that the researchers can supply their own datasets, define the types of each variable, and control the way the data is cleaned, filtered and handled. This includes that they can control which rotations are applied to each joint in the movement animation. On top of this, there should be more customizability in some of the plots, for example zooming in the map plot, loading a custom mesh in the movement plot, or coloring the embedding based on each possible variable. Another big challenge for future work to generate meaningful embeddings for heterogeneous time-varying data with missing values. For each variable type, another approach to the distance calculation is required to make it meaningful. A study of distance measures is needed. A though variable type to compare is for example the marker positions (posture information). Comparing posture automatically would not be straightforward, since you need to synchronize the movements in time. One person may be doing an action at another point in the dataset than another person. Misaligned movements create differences in distance measures between datasets. You might want to compare how both persons perform a certain action, or what both persons do at a certain time, or both those options. These options all require another approach, and research has to be done on what option gives a desirable result.

We believe that the developments in this thesis, despite being preliminary show the potential of visual analytics for sports data analysis.
A guide to using the visual analysis tool

This chapter provides a manual on how to perform some key steps in the visualization tool, and explains how the tool will behave in different cases. For technical specifications of the tool and its visualizations, you should read the research paper.

A.1. Selecting datasets

When you open the tool, you'll see a menu on the left side of the page. In the menu there are two folders: 'Embeddings' and 'Datasets'. Each of these folders opens a way to select your data.

A.1.1. Exploring

When you open the datasets folder, you will see a couple of folders of different sports, and an option to 'unselect all datasets'. When opening one of the sports folders, you will see all the datasets for that specific sport. Each dataset takes two rows in the menu. One row with the name, which will load the dataset when clicked on, and another row with the color of the dataset, that can be adjusted using the color picker. Each participant has its own color, and each dataset will have a different tone.

Loading a dataset will change the layout of the screen to the layout tailored for the chosen sport. In the menu you will see the dataset entry will be updated with the performance measure, calculated from the dataset. Also there is an indication that the dataset is currently selected. By selecting multiple datasets you can compare them to each other.

When picking a sport by selecting a dataset, the menu will also be tailored to that sport. The datasets for other sports disappear, and other menu entries appear. These will all be described in the next sections.

A.1.2. Embeddings

Another way to select datasets, is via embeddings. For a detailed explanation of what embeddings are and how they are created, you should read the paper about the visual analysis tool. In short, embeddings are two-dimensional representations of the collection of datasets, that can help you find similar datasets and hopefully identify clusters of good performances.

When opening the embeddings folder, you see an option to create a new embedding or to pick one from a list of previously created embeddings.

For creating a new embedding, you first have to select the sport. Once you have done this, a list of datasets will appear, where you can select which datasets should be in the embedding. Below that, there is a list of variables with sliders next to them, where you can set how much
A. A guide to using the visual analysis tool

Each variable weighs in the embedding. You can set a default value for all variables, or when the variables are grouped (as in baseball or speed skating), you can give each group a certain value.

When you click the create button, the embedding will be generated. This can take quite some time, in the order of minutes if you have selected more than ten datasets. When the embedding is calculated, it will be added to the list of previous embeddings, and it will be loaded, changing the layout of the screen. The embedding plot will be shown and by dragging a selection box around some of the circles you can select datasets. More information on how to interpret the embedding will be given in a later section.

A.2. Using the plots
Once you have selected a sport, the layout of the screen changes to be able to show as much information as possible for this specific sport. Several plots may appear on the screen. For the technical part behind each plot you are redirected to the paper about the visual analysis tool. Now we will go through all the plots and learn about what we can do with them and how to do it.

A.2.1. Line plot
The line plot shows the curves of selected variables, in the color of each dataset. There is a legend of the selected variables, and hovering over it highlights all lines of that variable. Clicking anywhere in the plot shifts the animation to that frame. In the 'Line Plot' menu entry, you can remove all variables; choose to normalize the data, so that each variable has a range from 0 to 1 (useful for comparing different variables to each other); Choose to display all datasets, only the plain datasets, or only clusters (more on clusters later); you can change the range of the X axis, allowing you to inspect a certain part of the data better; you can choose to display the magnitudes of vectors as their separate components; and finally you can change the background color of the plot.

Selecting variables to display in the line and vector plots can be done from the 'variables' menu, that is added to the menu at the bottom. Clicking a variable once will add the variable to both plots, clicking it again will remove it. An indicator will be shown in the variable list for all selected variables. Variables can also be removed by right-clicking the variable name in the legend of the line plot. The same applies for removing datasets, this can be done by right-clicking a line in the plot.

A.2.2. Vector plot
When there are three-dimensional variables in the dataset (such as moments or angles), these will be visualized as a vector in a sphere, indicating the direction of the variable. At the same time, the magnitude of the variable will be shown in the line plot. The vector plot can be rotated, and when the animation is running, the arrow will move, leaving behind a trace that can be customized by using the settings in the 'Vector plot' menu entry. You can customize the amount of ghost steps left behind, and whether the variance, that is displayed as a circle in the case of a cluster dataset, is also leaving a ghost trace.

Selecting variables is done the same way as for the line plot.

A.2.3. Animation
If your chosen sport has an animation, you will see a 3D object on the screen. In the "Animation" menu, you can customize a lot about what you want to see or hide in this view. There is again a possibility to let the object leave a trace, and you can even record a GIF image of a movement. Apart from that, you can control the animation itself. When clicking animate it will
start moving, and stepping and stepping back can help you inspect the movement closer. All the other plots will be synchronized with the animation. You can move the camera around as you wish by interacting with the screen with your mouse.

A.2.4. Trajectory plot for cycling
When cycling is the sport you have chosen, you will see a plot of a map, with the track ridden by the cyclist in its respective dataset color plotted on the map. When the animation is on, a circle indicating a rider will move over the map. When the circle gets an extra border, this means that the rider is braking at that moment. Clicking on one of the hectometer signs moves the animation to that point. When you have selected a certain range in the line plot, the start and finish location of the range will be shown in the map plot as well. The map plot has an option to toggle between sampling the data on distance (in meters), as default, or on time. If you sample the data on time, you will be able to see the descend as if the riders are racing against each other, instead of seeing them all at the same point.

A.2.5. Trajectory plot for speed skating
When speed skating is the sport you have chosen, you will see a small plot showing the position of the skater in its stroke on the track. This plot can show you how stroke sizes of different skaters relate to each other. There are no interactions possible with this plot.

A.2.6. Event plot
When baseball is the sport you have chosen, you will see a plot with some circles on it. Each circle indicates an event happening at that point in time. Clicking the circle will move the animation to that frame. This can help you spot interesting differences between pitchers.

A.2.7. Embedding plot
In the embedding plot you will see a number of dots, equal to the amount of datasets in the embedding. Each dot represents one dataset. You can use the legend tooltip to find out which datasets are included in the embedding, plus the weights of the variables that are used to calculate the embedding. You will also find a legend for the coloring of the participants and the performance, which you can change from the ‘Embedding’ menu. Coloring on performance can give you valuable information about clusters with a ‘good’ performance, having a lighter color. Selecting datasets can be done by dragging a selection box around them, hold ctrl and drag to add datasets to your selection. Selected datasets will have a border around the circle in the color tone of the dataset.

A.3. Clusters
Once you have selected several datasets, it can be useful to save them as a cluster. You can then compare this cluster to other clusters, and that may help you find the crucial difference between a regular performance and ‘the best’ performance. To create a cluster, use the ‘cluster’ menu. Here you can give the cluster a name, view the existing clusters, and check which datasets are in a cluster. If you create multiple clusters, the clusters will be sorted in their performance, which is reflected in the cluster color: yellow for the best cluster, to red for the worst.

Clusters will behave as normal datasets in most of the plots, with some small differences. In the line plot, clusters will show the standard deviation of each variable with a transparent curve, and the same happens in the vector plot, in the shape of a circle. In the animation, the most representative dataset will be visualized, and the other datasets are present as ‘ghosts’. In the embedding plot, datasets that are in a cluster will get an extra ring around them in the

Clusters will behave as normal datasets in most of the plots, with some small differences. In the line plot, clusters will show the standard deviation of each variable with a transparent curve, and the same happens in the vector plot, in the shape of a circle. In the animation, the most representative dataset will be visualized, and the other datasets are present as ‘ghosts’. In the embedding plot, datasets that are in a cluster will get an extra ring around them in the
color of the respective cluster. In the line plot there is also an option to display only the clusters. If you select this option, the best cluster will be chosen as baseline, and for all other clusters the difference to this baseline will be visualized.
Implementation details

In this appendix we describe how the software is structured, and how you should be able to run the tool. We start with explaining what software is used to develop the tool, then we describe how the source code is structured, and finally we explain how to run the tool.

B.1. Software

The visual analysis tool is largely written in javascript. The choice for javascript was made because a javascript program can easily be opened in the browser, and does have special requirements. Also javascript has plenty of plugins that can be used, allowing us to focus on the novel things.

We have used several javascript plugins to create the visualization tool:

- **d3** - D3 is used to create different plot throughout the tool, such as the line plot, event plot, position plot and embedding plot.
- **dat-gui** - dat.gui is used for the menu of the tool.
- **jquery** - jQuery is used almost everywhere.
- **js-logger** - JS Logger is used for debugging and for sending notifications to the user.
- **lodash** - Lodash is a library that makes working with arrays, numbers, objects, and strings much easier.
- **three** - Three.js is a WebGL plugin that is used to display the movement plot and the vector plot.
- **tooltipster** - Tooltipster is used to display tooltips for several plots, for example the legend of the embedding plot.
- **canvg** - canvg is used to render the svg plots to a HTML canvas and save a screenshot.
- **CCapture & whammy** - These tools are used to save and export a GIF image of the movement plot animation.
- **Google Maps** - The Google Maps API is used to display the map for the GPS plot. Several functions are used to plot lines and images on the map.
Some server-side code is written in PHP, since I was familiar in this language for writing server-side code. This is limited to the ‘ajax’ folder (see next section for explanation on folders).

The algorithm to calculate new embeddings is written in python. This is because this allowed us to use the powerful numpy plugin, that is very fast at doing calculations with big arrays of numbers. These python files are called from a PHP script that in turn is called from the javascript front end.

B.2. Source code structure

When you are looking at the root folder of the project, you see several folders. The most important folders and their functions and content will be discussed here.

B.2.1. Ajax

The ajax folder handles the ajax requests to the server. This includes fetching a list of the available datasets and embeddings, uploading new datasets, and creating and deleting embeddings.

B.2.2. Data

In the data folder created embeddings are saved, as well as all of the datasets. For each type of dataset, there is a separate folder. These folders are named by the sport they represent, followed by an optional extra name (e.g. ‘Strokes’ for the speed skating strokes datasets).

B.2.3. Plugins

The plugins folder holds some javascript plugins, more specifically, the plugins that were not available via npm. There is also a file, generically called ‘plugin.js’, that is used to extend basic javascript and other plugins with functions that were missing for this project. The dat.gui and three.js plugins have received the most extra functions. These plugins are used for the menu and for the 3D visualizations.

B.2.4. Scripts

The scripts folder forms the basis of the visualization tool. An object-oriented approach is chosen for the javascript application. There are classes for the VisualizationTool, Data, Directory, Dataset, Menu, Plot (and subplots), Embedding and Cluster. Each class has its own file, and there are some additional files that define global or more specific logic.

- **app.js** - This file defines the controller of the visualization tool, saved as global variable VISTOOL. This controls the data and plots, can send notifications, and has a link to the menu.

- **_layout.js** - This file contains a layout function that is called when the layout has to be updated (i.e. on a data type change). This function handles the sizing, showing and hiding of different plots.

- **data.js** - This file controls the data of the tool. This includes loading the list of datasets, selecting and unselecting datasets, uploading new datasets, selecting and unselecting variables and determining variable type, saving clusters and distributing dataset colors.

- **directory.js** - A Directory object is created for each folder in the data folder we have seen earlier. This way we can see what kind of data is displayed (only the subtype, since the main type is handled by the VISTOOL).
• **dataset.js** - A Dataset object is created for each file in each folder in the data folder. The participant of the dataset is identified according to the file name, as well as the type, according to the folder it is in. The dataset can be loaded, after which it will contain the actual data, sorted by time step. There are several methods to check whether a variable is available in the dataset, and to calculate the minimum, maximum, mean and range of a variable. You can also get the magnitudes of a variable, the standard deviation, and the values of a single variable in all time steps. You can get the time steps in the dataset, and the performance measure. You can also calculate the distance to another dataset, which is used for calculating the most representative dataset in a cluster. There are some functions for formatting the dataset name that is displayed in the tool. You can take a subset of the dataset, sample the dataset to a fixed amount of time steps, and save the dataset to a csv file.

• **menu.js** - The menu creates a dat.gui instance, that functions as the menu for the tool. The menu has entries for all relevant plots that can be customized, and it allows for dataset and variable selection. You can select existing embeddings and create new embeddings from the menu as well. The menu is kept up to date by various methods. Dataset and variable selection are always displayed because the controllers that display these in the menu are saved on the dataset object, so that changes can be reported to these controllers immediately.

• **plot.js / plot folder** - Plot is the basic object from which most plots inherit their functions. It defines methods that control the size and position of the plot. The background color of the plot can be set, and there are generic update, clear, and clickListener methods. There is also a method that allows you to save a screenshot of the plot. The update function is the method that does the rendering of the plots, and therefore the most used and biggest function.

  – **plot-line.js** - The line plot has some methods to change the time range, this is primarily used to be able to switch between meters and seconds in the cycling datasets, but can be extended to more variables. There is a normalize function that is used to normalize the lines between 0 and 1, to be able to compare different variables to each other. There is a function to set the reference cluster, used in the cluster view, and there are some listeners for clicking and animations. The update method contains all the logic for the plotting.

  – **plot-vector.js** - The vector plot updates when the selected variables change, and creates a vector subplot for each vector variable. These subplots are updated every animation frame. There is a method that creates a joint that is displayed in the center of the sphere, which needs improvement (see future work).

  – **plot-movement.js** - The movement plot is the only plot that does not inherit from the parent Plot. Therefore, it contains functions to manage the size and position of the plot. There are functions to reset the camera, and to change the background color. The most important functions are the ones that control the animation: stepping, animating, pausing and resetting. In addition to that, there are functions to add and remove datasets and clusters, and to update several types of meshes. The animation methods call to several other methods, that are defined in files in the plot-movement folder. There is a separate file for each type of mesh.

  – **plot-gps.js** - The GPS plot has some methods to initialize Google Maps and plot the lines, start and finish, and the hectometer indicators. There are also functions to
move these markers, and functions to add and remove datasets. Another function is to toggle the time variable (meters / seconds)

- **plot-embedding.js** - This file defines functions to define the embedding points and files, and to display the embedding and the embedding legend.
- **plot-position.js** - The position plot is a line plot that is automatically created when a skating dataset is loaded, and updated when the animate event is fired. The logic for this is contained in this file.
- **plot-event.js** - This file contains the update method and a method that listens to the animate event, that moves the line to the animated frame.

• **cycling.js** - In this file, an event listener is initiated that calculates the time difference for all loaded cycling datasets.

• **speed-skating.js** - This file defines the functions for extracting and saving stroke datasets from complete datasets.

• **embeddings.js** - This file contains the logic for loading existing embeddings and creating new ones.

• **cluster.js** - This file defines the functions to create a new cluster, calculate the average of the containing datasets, determine the most representative dataset of the cluster, and determine the colors of the existing clusters.

### B.3. Running the tool

#### B.3.1. Requirements
To run the local web server, several programs should be installed and added to the PATH variable:

- PHP (>= 5.5)
- Python (including numpy(!) for calculating new embeddings)
- Node.js

To get a local copy of the web server, you should clone or download the git repository at https://github.com/RalfNieuwenhuizen/vistool.

In this repository, install all dependencies once via npm from the command line: npm install.

Afterwards, running the web server is as easy as running the command grunt server from the command line.
List of variables

This appendix holds a list of all variables in the different datasets including an extended description of each variable.

C.1. Cycling
The cycling datasets contain measurement data of the measurement bike.

- Latitude of GPS position (lat)
- Longitude of GPS position (lon)
- Distance traveled in meters (s), this measure is derived from the GPS position
- GPS-derived speed in km/h (speed)
- Time traveled in seconds (t)
- Angle the steer makes with the frame (steer)
- Angle the bike makes with the ground (roll)
- Front brake “power” (fb), the extent to which the front brake is used.
- Rear brake “power” (rb), the extent to which the front brake is used.
- Front brake ratio (fbr), the ratio compared to the rear brake.
- Rear brake ratio (rbr), the ratio compared to the front brake.
- Comments left on the dataset about e.g. malfunction in sensor (comment)
- Derived measurement that gives the time difference between this dataset and the fastest in the selection (timeDiff)

C.2. Speed Skating
The speed skating datasets have measurement data from the instrumented skate and from the markers, measured by high speed cameras. For most variables the 'L' or 'R' in the name indicates the side of the body this measurement is for: left and right, respectively.
• Normal force, the force the skater applies to the ice in the direction he/she pushes. (LSFn, RSFn)

• Lateral force, the force the skater applies in a sideward direction to the ice. (LSFl, RSFl)

• Resultant global force (3D: sideways, forwards, upwards), the resulting force working on the skater. (LFG2, RFG2)

• Wide-range accelerometer (3D: wLACC, wRACC)

• Low-noise accelerometer (3D: LACC, RACC)

• Gyroscope (3D: LGYR, RGYR)

• Magnetometer (3D: LMAG, RMAG)

• Position data (3D), the global positions of different markers, all having an X (sideward displacement), Y (forward displacement) and Z (upward displacement) component. For an overview of the markers, see figure 2.2.
  – Body (B)
  – Body displacement (DB)
  – Derived position of the skate (S)
  – Outer ankle (FAL)
  – Big toe (FM2)
  – Small toe (FM5)
  – Rear skate tip (SKATE)
  – Outer knee (FLE)
  – Hip (ICT)
  – Upper shin (TTC)
  – Shoulder (SAE)

• Orientation data (of skates), roll is the leaning angle of the skate, yaw is the steering angle of the skate, and pitch is the lifting angle of the skate.
  – LSroll, LSyaw, LSpitch
  – RSroll, RSyaw, RSpitch

C.3. Baseball
The baseball datasets contain measurement data of several inertial measurement units placed on the body.

• Events, the time of events in baseball pitching in milliseconds
  – Time of foot contact (FC)
  – Time of maximal shoulder external rotation (MER)
  – Time of ball release (BR, the dataset is centered around this time)
  – Time of maximal shoulder internal rotation (MIR)
  – Time of maximal pelvis rotation velocity (MPW)
C.3. Baseball

- Time of maximal Thorax rotation velocity (MTW)

- Ball velocity estimate ($V_{\text{Ball}}$, in m/s). This estimation is done by using the velocity of the center of mass of the hand at the time of ball release.

- Linear velocity ($v$), the rate at which the body segments move
  - Linear velocity of the center of mass of the hand (Hand_com)
  - Linear velocity of the center of mass of the lower arm (LowArm_com)
  - Linear velocity of the center of mass of the upper arm (UpArm_com)
  - Linear velocity of the center of mass of the thorax (Thorax_com)
  - Linear velocity of the center of mass of the pelvis (Pelvis_com)
  - Linear velocity of the center of mass of the full body (COM)

- Linear acceleration ($a$), the rate at which the body segments accelerate
  - Linear acceleration of the center of mass of the hand (Hand_com)
  - Linear acceleration of the center of mass of the lower arm (LowArm_com)
  - Linear acceleration of the center of mass of the upper arm (UpArm_com)
  - Linear acceleration of the center of mass of the thorax (Thorax_com)
  - Linear acceleration of the center of mass of the pelvis (Pelvis_com)
  - Linear acceleration of the center of mass of the full body (COM)

- Joint angles (angles), the Euler angles of different joints in the body.
  - Joint angles of the thorax (Thorax_ZXY)
  - Joint angles of the shoulder (Shoulder_YXY)
  - Joint angles of the elbow (Elbow_ZXY)
  - Joint angles of the wrist (Wrist_ZXY)

- Angular velocity ($\omega$), the rate at which body segments and joints rotate.
  - Angular velocity of the hand (Hand)
  - Angular velocity of the lower arm (LowArm)
  - Angular velocity of the upper arm (UpArm)
  - Angular velocity of the thorax (Thorax)
  - Angular velocity of the pelvis (Pelvis)
  - Angular velocity of the wrist joint (Wrist_ZXY: joint angular velocity, ZXY means that this is the angular velocity projected on the joint coordinate system)
  - Angular velocity of the elbow joint (Elbow_ZXY)
  - Angular velocity of the shoulder joint (Shoulder_YXY)

- Joint moment (Moment_J), the moments working in the different directions on the joints.
  - Net joint moment in wrist joint (Wrist_ZXY)
  - Net joint moment in elbow joint (Elbow_ZXY)
  - Net joint moment in shoulder joint (Shoulder_YXY)
Transformations

To reproduce the movement of the bodies from the marker data in the speed skating and baseball datasets, the data had to be fitted to the model of the mesh that was used. For both sports, the data was formatted in a different way, which led us to using two different approaches, one more successful than the other. Both approaches will be discussed here.

Simply applying the data to the mesh resulted in some funny transformations, as you can see in figure D.1. To apply the correct transformations, the data had to be converted to match with the coordinate system of the mesh. There is a standard reference frame that is used to measure the data for both sports. More information on this standard can be found in Wu et al. [14][15]. The mesh that was used for the visualization unfortunately did not follow this standard.

![Figure D.1: Transformations do not always give the desired result.](image)

For cycling, the roll and steer angles from the data were simply applied to the respective axis of the cycle model as a rotation. It is hard to verify this transformation, since the data is not clean nor filtered. When there is more data available, this transformation requires more attention.

**Speed skating: creating rotation matrices from global marker positions**

The skating data contains the global positions of several body markers. Since the mesh is a chain of several bones, traversing the body in distal direction (outward) starting from the spine, each bone is rotated relative to its parent.

From the global positions of the body markers, we could calculate the orientations of the different bones, that define the rotation matrices of the joints relative to their parent. Since this
matches the way the mesh works, these relative rotations could be applied to the bones.

In practice, this means the following joints are defined, for which the relative rotations are calculated.

- **Back**: from 'FLOOR' via 'SACR' to 'TV2',
- **Hip**: from 'SACR' via 'ICT' to 'FLE',
- **Knee**: from 'ICT' via 'FLE' to 'FAL',
- **Ankle**: from 'FLE' via 'FAL' to 'FM5'.

A help vector is created from the 'from' marker to the 'via' marker, and the y axis is defined from the 'to' marker to the 'via' marker. The x axis is defined as the cross product of these two vectors. If these two vectors are in line with each other, the x axis will not be defined properly. To solve this problem, the x axis from the previous rotation is used. On top of this, a check is done whether there is a direction change of the x axis relative to the parent, a if this is the case, the x axis is inverted. A complete rotation of a limb is not possible for humans, so this would indicate a fault in the data. Finally, the z axis is calculated as the cross product of the x and y axes. These vectors form the rotation matrix of the bone in the global coordinate system.

Finally, the rotation of a bone is set by pre-multiplying the calculated rotation with the inverse of the world matrix of the parent bone. This way, you set the relative rotation of the bone with respect to the parent in the global coordinate system.

**Baseball: adjust local coordinate systems to model coordinate systems**

For the baseball data, where the data was defined in Euler angles, there was another problem. Since the coordinate systems in which the data was measured were different from the coordinate systems of the mesh, a mapping had to be created that made the transformation from one coordinate system to another.

The data were measured in the standard coordinate system that was defined by Wu et al.[15]. After several brainstorm sessions with the baseball researcher and the daily supervisor, we concluded that the transformation $T$ that had to be done was constructed as follows.

$$
T = M_2 R M_1 \begin{pmatrix} x \\ y \\ z \end{pmatrix}^T
$$

$M_2$ is the transformation from the global coordinate system of the mesh to the local coordinate system of the mesh.

$R$ is the transformation from the global coordinate system of the data to the global coordinate system of the mesh.

$M_1$ is the transformation from the local coordinate system of the data to the global coordinate system of the data.

$\begin{pmatrix} x \\ y \\ z \end{pmatrix}$ is the rotation matrix created from the Euler angles in the data.

We have tried to apply this transformation to the bones, but had no success.

Eventually, we created a simple stick figure to which the transformations could be applied directly.
Evaluation forms
### Usability

<table>
<thead>
<tr>
<th>I would like to use this system to analyse my data</th>
<th>Strongly disagree</th>
<th>Strongly Agree</th>
</tr>
</thead>
<tbody>
<tr>
<td>I found the system unnecessarily complex</td>
<td></td>
<td></td>
</tr>
<tr>
<td>I thought the system was easy to use</td>
<td></td>
<td></td>
</tr>
<tr>
<td>I found the various functions in the system were well integrated</td>
<td></td>
<td></td>
</tr>
<tr>
<td>I found the system very cumbersome to use</td>
<td></td>
<td></td>
</tr>
<tr>
<td>I felt very confident using the system</td>
<td></td>
<td></td>
</tr>
<tr>
<td>I needed to learn a lot of things before I could get going with this system</td>
<td></td>
<td></td>
</tr>
<tr>
<td>With this tool I can understand the data better</td>
<td></td>
<td></td>
</tr>
<tr>
<td>I can discover new relationships and insights on the data using this tool</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Features

<table>
<thead>
<tr>
<th>Line plot</th>
<th>Useless</th>
<th>Useful</th>
</tr>
</thead>
<tbody>
<tr>
<td>GPS plot (position on the road)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Animation (movement reconstruction)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Embeddings (similarity of datasets)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Clusters</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Data selection</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Menu layout</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Linking between views</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Tasks

<table>
<thead>
<tr>
<th>Task 1</th>
<th>Irrelevant</th>
<th>Relevant</th>
</tr>
</thead>
<tbody>
<tr>
<td>Task 2</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**This tool is useful for:**

- ...
- ...

**I am missing the following features:**

- ...
- ...
### Usability

<table>
<thead>
<tr>
<th>Statement</th>
<th>Strongly disagree</th>
<th>Agree</th>
</tr>
</thead>
<tbody>
<tr>
<td>I would like to use this system to analyse my data</td>
<td></td>
<td></td>
</tr>
<tr>
<td>I found the system unnecessarily complex</td>
<td></td>
<td></td>
</tr>
<tr>
<td>I thought the system was easy to use</td>
<td></td>
<td></td>
</tr>
<tr>
<td>I found the various functions in the system were well integrated</td>
<td></td>
<td></td>
</tr>
<tr>
<td>I found the system very cumbersome to use</td>
<td></td>
<td></td>
</tr>
<tr>
<td>I felt very confident using the system</td>
<td></td>
<td></td>
</tr>
<tr>
<td>I needed to learn a lot of things before I could get going with this system</td>
<td></td>
<td></td>
</tr>
<tr>
<td>With this tool I can understand the data better</td>
<td></td>
<td></td>
</tr>
<tr>
<td>I can discover new relationships and insights on the data using this tool</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Features

<table>
<thead>
<tr>
<th>Feature</th>
<th>Useless</th>
<th>Useful</th>
</tr>
</thead>
<tbody>
<tr>
<td>Line plot</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sphere plot (vector visualizations)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Position plot (position on the track)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Animation (movement reconstruction)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Embeddings (similarity of datasets)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Clusters</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Data selection</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Menu layout</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Linking between views</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### This tool is useful for:

### I am missing the following features:

---
# Evaluation form Visualization Tool - Speed Skating

## Usability

<table>
<thead>
<tr>
<th>Strongly disagree</th>
<th></th>
<th>Strongly Agree</th>
</tr>
</thead>
<tbody>
<tr>
<td>I would like to use this system to analyse my data</td>
<td></td>
<td></td>
</tr>
<tr>
<td>I found the system unnecessarily complex</td>
<td></td>
<td></td>
</tr>
<tr>
<td>I thought the system was easy to use</td>
<td></td>
<td></td>
</tr>
<tr>
<td>I found the various functions in the system were well integrated</td>
<td></td>
<td></td>
</tr>
<tr>
<td>I found the system very cumbersome to use</td>
<td></td>
<td></td>
</tr>
<tr>
<td>I felt very confident using the system</td>
<td></td>
<td></td>
</tr>
<tr>
<td>I needed to learn a lot of things before I could get going with this system</td>
<td></td>
<td></td>
</tr>
<tr>
<td>With this tool I can understand the data better</td>
<td></td>
<td></td>
</tr>
<tr>
<td>I can discover new relationships and insights on the data using this tool</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

## Features

<table>
<thead>
<tr>
<th>Useless</th>
<th></th>
<th>Useful</th>
</tr>
</thead>
<tbody>
<tr>
<td>Line plot</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sphere plot (vector visualizations)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Position plot (position on the track)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Animation (movement reconstruction)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Embeddings (similarity of datasets)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Clusters</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Data selection</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Menu layout</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Linking between views</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

## Tasks

<table>
<thead>
<tr>
<th>Irrelevant</th>
<th></th>
<th>Relevant</th>
</tr>
</thead>
<tbody>
<tr>
<td>Task 1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Task 2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Task 3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Task 4</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Task 5</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**This tool is useful for:**

**I am missing the following features:**
**Evaluation form Visualization Tool - Baseball**

### Usability

<table>
<thead>
<tr>
<th>Statement</th>
<th>Strongly disagree</th>
<th>Strongly Agree</th>
</tr>
</thead>
<tbody>
<tr>
<td>I would like to use this system to analyse my data</td>
<td></td>
<td></td>
</tr>
<tr>
<td>I found the system unnecessarily complex</td>
<td></td>
<td></td>
</tr>
<tr>
<td>I thought the system was easy to use</td>
<td></td>
<td></td>
</tr>
<tr>
<td>I found the various functions in the system were well integrated</td>
<td></td>
<td></td>
</tr>
<tr>
<td>I found the system very cumbersome to use</td>
<td></td>
<td></td>
</tr>
<tr>
<td>I felt very confident using the system</td>
<td></td>
<td></td>
</tr>
<tr>
<td>I needed to learn a lot of things before I could get going with this system</td>
<td></td>
<td></td>
</tr>
<tr>
<td>With this tool I can understand the data better</td>
<td></td>
<td></td>
</tr>
<tr>
<td>I can discover new relationships and insights on the data using this tool</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Features

<table>
<thead>
<tr>
<th>Feature</th>
<th>Useless</th>
<th>Useful</th>
</tr>
</thead>
<tbody>
<tr>
<td>Line plot</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sphere plot (vector visualizations)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Event plot (timing of events)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Animation (movement reconstruction)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Embeddings (similarity of datasets)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Clusters</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Data selection</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Menu layout</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Linking between views</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Tasks

<table>
<thead>
<tr>
<th>Task</th>
<th>Irrelevant</th>
<th>Relevant</th>
</tr>
</thead>
<tbody>
<tr>
<td>Task 1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Task 2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Task 3</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**This tool is useful for:**

**I am missing the following features:**
Bibliography


