NBA Player Statistics Visualization

https://github.com/wilsonCernWq/NBAstatsVIS.git

Qi WU qwu@cs.utah.edu (u1077116)
Mengjiao HAN u1078078@utah.edu (u1078078)
Qiuhua SHENG jessica.sheng@utah.edu (u0856465)

1. Overview and Motivation

As more and more accurate recording technologies are involved in modern basketball games, analyzing and visualizing basketball players' performance based on enormous statistics has become a hot topic. With the help of a great visualization tool, measuring players' abilities and comparing different players' performance is not difficult. There are a lot of tools and websites based on NBA players' statistical purpose. However, most websites include excessive data which can be understood by non-technical people. As basketball enthusiasts, we want to create a player profiler for basketball fans to search interested player's performance and compare different players in a more efficient method.

2. Related Work

There are plenty of statistics websites for NBA players, such as the NBA official website (http://stats.nba.com), Basketball Reference (http://www.basketball-reference.com), Fox Sports, Yahoo Sports and so on. Users can search for various kinds of raw data as they are shown in below. Although raw data could be essential for professional sport analysts, it is not interesting for an ordinary basketball fan or sport fan in general. It is not an efficient way for general users to understand and compare the information. Therefore, we think having a good visualization of those data is crucial.

PLAYER STATS TOTALS PER GAME

REGULAR SEASON +

Season	TEAM	AGE	GP	GS	MIN	PTS	FGM	FGA	FG%	3PM	3PA	3P%	FTM	FTA	FT%	OREB	DREB	REB	AST	STL	BLK	TOV	PF
2016-17	DET	23	8	8	30.4	14.4	5.8	12.0	47.9	0.0	0.0	0.0	2.9	5.5	52.3	3.3	11.0	14.3	1.6	1.4	1.4	2.1	2.5
Career:			312	262	29.7	13.2	5.7	10.3	55.1	0.0	0.0	25.0	1.8	4.8	38.4	4.7	7.9	12.6	0.6	1.2	1.6	1.5	3.1

Figure 1 Example of NBA Player Statistics from NBA Official Website

Last 10 Games				FG			3PT			FT			Reb.			Misc					
Date	Opp.	Score	Min	FGM	FGA	FG%	3PM	3PA	3PT%	FTM	FTA	FT%	Off	Def	Reb	Ast	TO	Stl	Blk	PF	Pts
Nov 9	@PHO	L 100-107	35:05	7	13	53.8	0	0	0.0	4	6	66.7	4	10	14	4	2	1	1	5	18
Nov 7	@LAC	L 82-114	24:52	6	10	60.0	0	0	0.0	3	5	60.0	4	8	12	0	3	2	1	3	15
Nov 5	DEN	W 103-86	33:55	6	10	60.0	0	0	0.0	7	11	63.6	3	17	20	1	4	1	3	2	19
Nov 2	@BKN	L 101-109	25:10	3	9	33.3	0	0	0.0	0	0	0.0	0	6	6	4	2	0	2	2	6
Nov 1	NY	W 102-89	35:18	4	13	30.8	0	0	0.0	1	6	16.7	2	11	13	2	1	2	1	1	9
Oct 30	MIL	W 98-83	36:48	8	16	50.0	0	0	0.0	4	8	50.0	8	15	23	1	0	2	3	1	20
Oct 28	ORL	W 108-82	27:34	5	14	35.7	0	0	0.0	2	2	100.0	4	16	20	1	1	2	0	1	12
Oct 26	@TOR	L 91-109	24:09	7	11	63.6	0	0	0.0	2	6	33.3	1	5	6	0	4	1	0	5	16

Figure 2 Example of NBA Player Statistics from Yahoo Sport Website

There are good visualizations of basketball data already, for example the work done by Peter Beshai. However, his website only focuses on displaying spatial data in a fancy way, without providing comprehensive schemes for interpreting player's performance and ability based statistics. This encourages us to redesign a visualization of NBA player statistics.

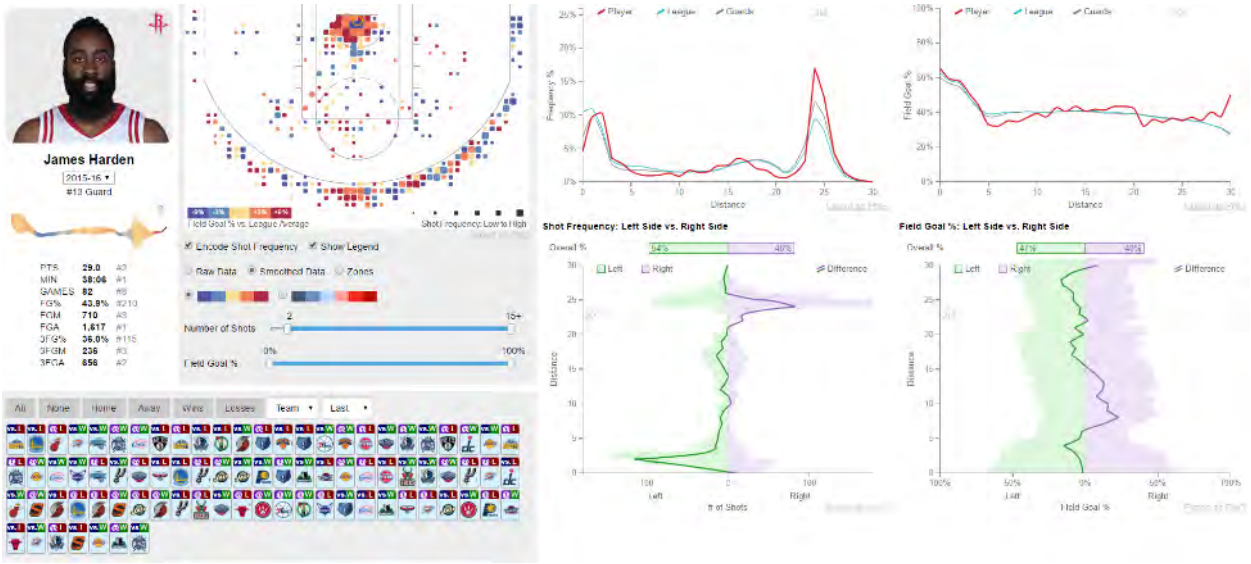

Figure 3 Peter Beshai's Visualization http://buckets.peterbeshai.com/app/#/playerView/201935_2015

3. Target Users and Task

The target users of our design are basketball enthusiasts, who would be interested in knowing a basketball player's performance from statistics without understanding excessive professional statistical attributes. In our design, a user should be able to search for the player interested in and select proper time ranges. Probably we will provide comparison tools also for data exploration. The tasks of our design can be summarized as the following:

Analysis Task:

- Display players' basic information including image, name, weight and height, etc.

- Display players' basic performance data, such as points, assists, rebounds, field goal attempted, and turnovers.
- Display data details based on users' focus, for instance, displaying number of points made for each year or game within the given period.
- Display players' ranking of an attribute.
- Display data on both time base and spatial base, for example, scoring data in terms of both time and position on court.

Comparison Task:

- Compare two different players' performances
- Compare two different teams' performances

4. Questions

In our project, we are trying to analyze basketball players' performance. The first question we want to answer is related with simple date derived. For example, " What is A player's points in B time period?" or " How many is A player's assists in B time period?". Moreover, we want to analysis the balance of a player's skills. What is more, answers for the detail of a specific aspect, such as points, assists, rebounded, should be included in our design. For example, we need to derive data detail of rebound in order to let users to analyze or compare.

Based on previous questions we desire to answer in our design, we also come up with a new idea to show data as both time based and spatial based. Therefore, with analysis the data on time line, users could also analysis data by position of basketball yard.

5. Must Have Features

- Filter different players, year.
- Display players' basic information.
- Display player's team transfer.
- Derive data to show general players' points, defense rebounds, assists, offense rebounds and show the balance of players' performance.
- Derive data to show the ranking of a specific player based on the users' selecting aspect.
- Derive data to show the detail data of each aspect of a player.
- Compare two players' performance.

6. Optional Features

If we have extra time and mankind before the final submission, we will also implement a tool for customizing new teams based on users' selections. Users will be able to view and compare the teams they just created.

7. Data and data process

Data Source:

We will collect data from the API provided by www.nba.stats.com. The API generates JSON files based on query parameters we provided. We plan to include data in terms of both each player and each team. Player data consists of mainly four parts: player's general information, player's career summaries, player's season summaries and player's play-by-play records. All data has similar structures like the following figure (generated by http://jsonviewer.stack.hu).

Figure 4 Raw Data Structure Generated by http://jsonviewer.stack.hu

Scraping Method

We wrote python scrapers for grabbing data from the website. The program starts by downloading the player list, and then it will grab data for each player based on the list. All scripts we used have similar structures, so we will just show one example script here (generated by PyCharm, see python folder for the actual codes).

```python
import urllib
import os
import json


def mkdir(directory):
    if not os.path.exists(directory):
        os.makedirs(directory)


# open file
f = open('playerID-full.json', 'r')
data = f.read()
f.close()
data = json.loads(data)
mkdir('playerIcon')
os.chdir('playerIcon')
for entry in data['rowSet']:
    playerid = entry[0]
    print(str(playerid) + ' ' + entry[4])
    urllib.urlretrieve('http://stats.nba.com/media/players/230x185/' + str(playerid) + '.png', str(playerid) + '.png')
```

Figure 5 Python Scraping Script for Players' Icon

Cleanup

To effective reduce the total data size, we did a lot of data cleanup work. For example, we group multiple datasets into a single file and removed redundant/overlapping information. Here we displayed (part of) the compressed data structure.

```
|__ info
|       |__ PERSON_ID
|       |__ FIRST_NAME
|       |__ LAST_NAME
|       |__ BIRTHDATE
|       |__ SCHOOL
|       |__ COUNTRY
|       |__ HEIGHT
|       |__ WEIGHT
|       |__ SEASON_EXP
|       |__ JERSEY
|       |__ POSITION
|       |__ ROSTERSTATUS
|       |__ TEAM
|       |__ FROM_YEAR
|       |__ TO_YEAR
|       |__ DLEAGUE
|       |__ ALL_STAR
|
|__ career
|       |
|       |__ headerRow
|       |     = ["GP", "GS", "MIN", "FGM", "FGA", "FG_PCT", "FG3M",
|       |        "FG3A", "FG3_PCT", "FTM", "FTA", "FT_PCT", "OREB",
|       |        "DREB", "REB", "AST", "STL", "BLK", "TOV", "PF", "PTS"]
|       |
|       |__ headerCol = ['PerGame', 'Totals']
|       |
|       |__ RegularSeason = [ ${Career Data per Game}, ${Career Data Totals} ]
|       |
|       |__ PostSeason    = [ ${Career Data per Game}, ${Career Data Totals} ]
|
```

Figure 6 Cleaned Data Structure Example

However, only compressing datasets is not what we need to do all. To achieve the goal of querying attributes' spatial distributions, e.g. the shot chart, we will need to compute statistics like density function in the runtime, based on the query variables provided by users. We will in general use algorithms agree with common data processing tools such as MATLAB. There won't be outstanding (uncommon) mathematical techniques behind the scene.

8. Exploratory Data Analysis

Our data structures for each player's data can be found on "player.txt". The dataset type could be described as table. Our data types are attributes, items and include position. For each player, we have all detail data for each season year, such as points, the number of game play, and assists, etc. Moreover, each player has a player id. The data has hierarchical structure. The simplest visualization for our data is table or a bar chart. And interact them by change over time. Take the number of assists for example, we could use table or line chart to display it directly based on year. However, although our data is not complex, it is not a small database. Therefore, we also need zoom interaction to display detail.

Moreover, to visualize table in high-dimension, we could use an example from lectures, as figure 8.1 shows that. Therefore, for each attribute of a specific player, we could show the trance of data on different years. Also, because the data is dense, different color values should be used in our design. Based on our tasks, using data properly show the balance of players' skill is also important. For this purpose, we also need to rank data on different attributes to know the player's performance balance. Also, we could use color value to show the trend of specific attributes.

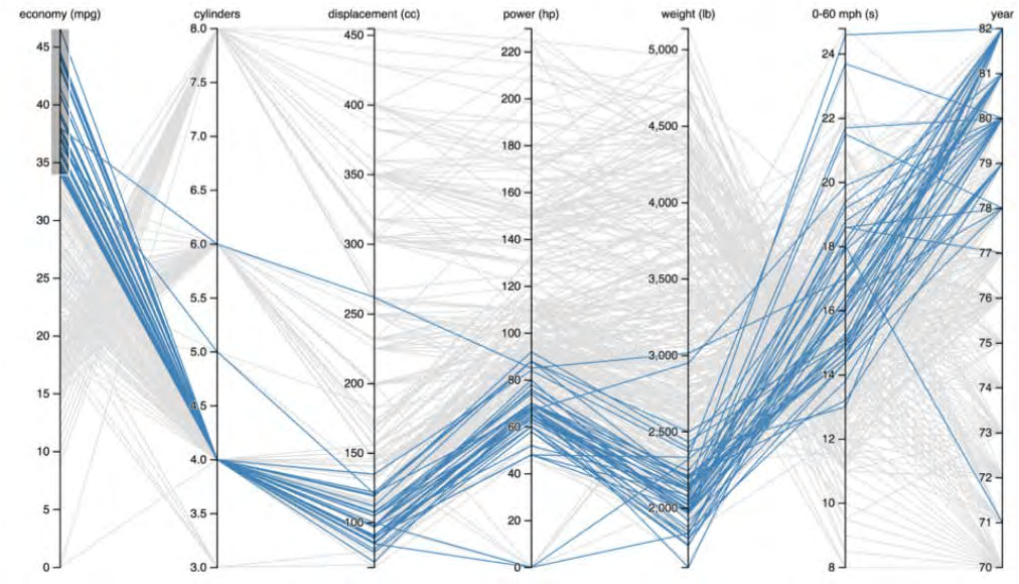

Figure 7 An example for Visualization of Table Dataset Type with Parallel Coordinate

Furthermore, every attribute's data from datasets is interval quantitative data type. But the scale of each attribute is not the same. Therefore, our design should have to consider the scale of attributes if we want to display several attributes together. At last, our data types include position data. For example, we know the point a player scores and know the position of scoring from our data. Therefore, we could use a map, which is a play field in our project, to visualize data by position.

9. Design Evolution

Different Views Design

As our tasks stated above, at the beginning, we design a visualization for general players' performance. As figure 8 shows that, we design four different views to show a player's general performance. The first one is a radar chart. On radar chart, each angle represents one aspect, for example, points, assists or turnover, etc. Another two simple visualizations are table and bar chart. The last one is ranking line chart. We think ranking line chart is too complex to our data structure. We need to search for data of different attributes and then rank them. Therefore, we prefer to choose radar chart or bar chart. By using a radar chart, user could know the balance of the player's skill easily. However, because the different aspects have different scales. It would be a problem to use same length of each attribute and might confuse users. Therefore, we decide to use bar chart to show these general data at first.

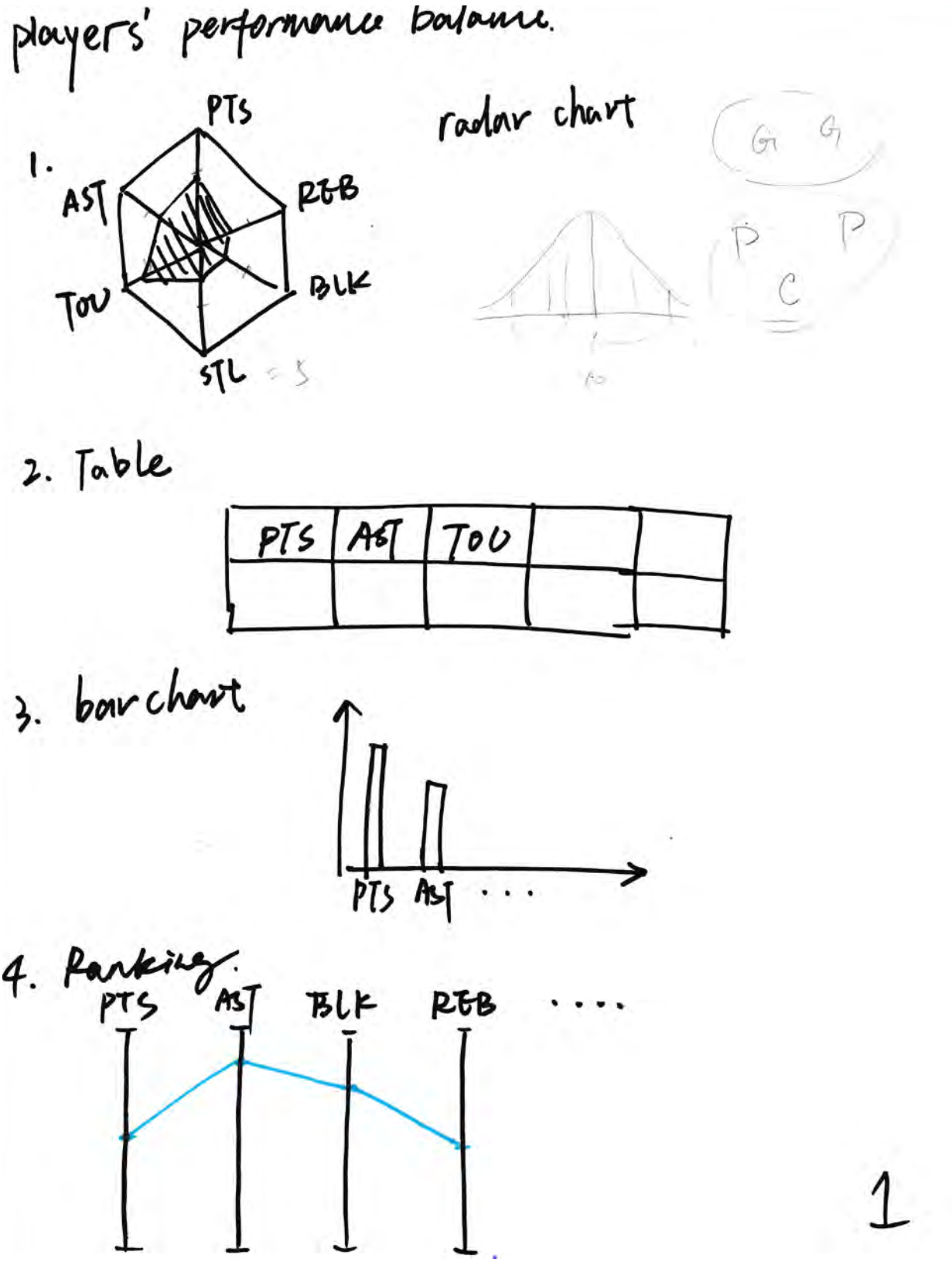

Figure 8 Designs for player's performance balance

Furthermore, to visualize team transfer. We designed map, time line and text. Map design would occupy too much space and geography position is not an important information. Therefore, we would use time line visualization. Another benefit of time line is that we could also add a brush here to let users choose a period.

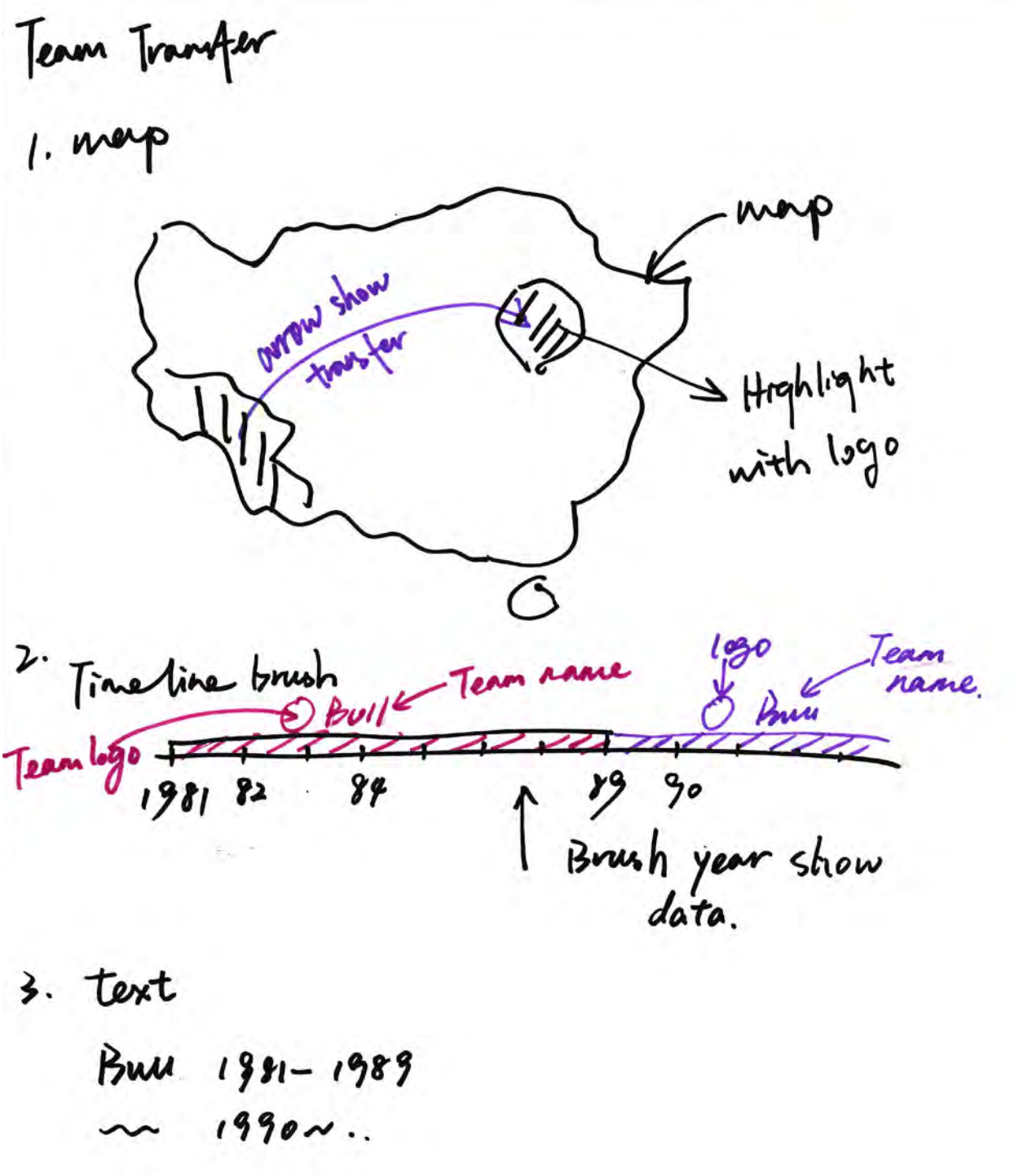

Figure 9 Designs for team transferring

Thirdly, we designed 3 different views for date detail visualization. When users click on an attribute from the general performance chart, for example, points. The detailed data would display. The first view could visualize data with position. So, we could have given user more direct insight of the position of a player scoring. Moreover, the value of points in one position would be represented by different colors. We would use color values, instead of hues, to show the data values. Another view is a line chart presents data with time. Because data would be dense, we would add a brush on the x axes to zoom detail. Moreover, another view is a map like heat map. For every year, the total number of games is similar. We are trying to visualize specific attribute data for each game in different years by using color values to show data values. This part is on figure 10.

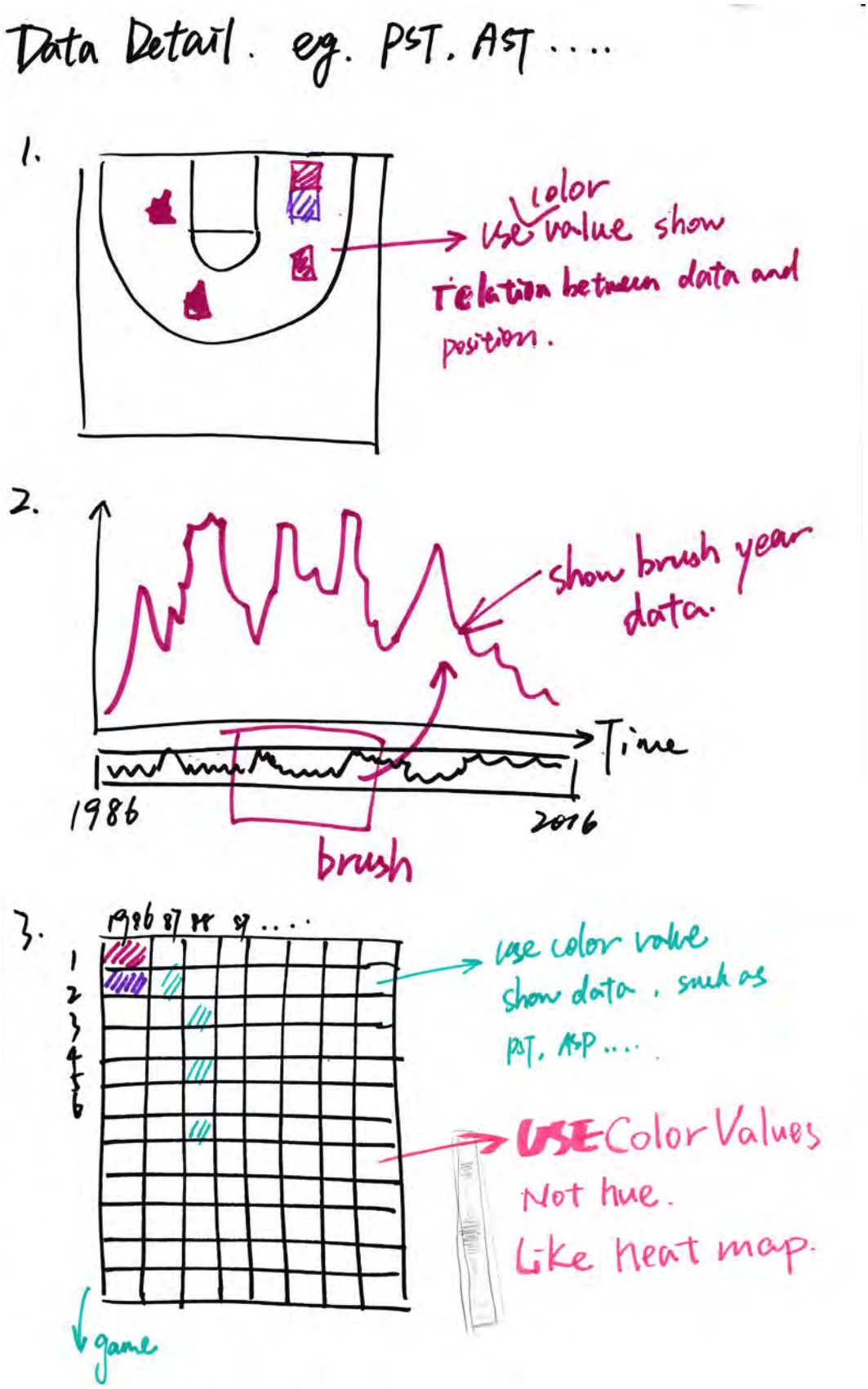

Figure 10 Designs for data details

Besides, we designed a ranking line chart. Because for a non-technical person may not understand meaning of a number. For example, he/she may not understand whether 20 points is excellent or common for a player. For this purpose, we design two ranking charts for each year, and connect them as figure 11 shows that. On figure 10, these two views are similar. In the first one, we use rectangles to show the real values of the attributes and the ranking. In the second, we just use ranking and use a tooltip to show the real number. However, because there are around 200 to 300 players, parallel coordinate could cross a lot and may hard to distinguish by users. Moreover, in both views, we would use brush to display the players' name for users to know the ranking detail. So far, we choose the first view, but we are still looking for a new version.

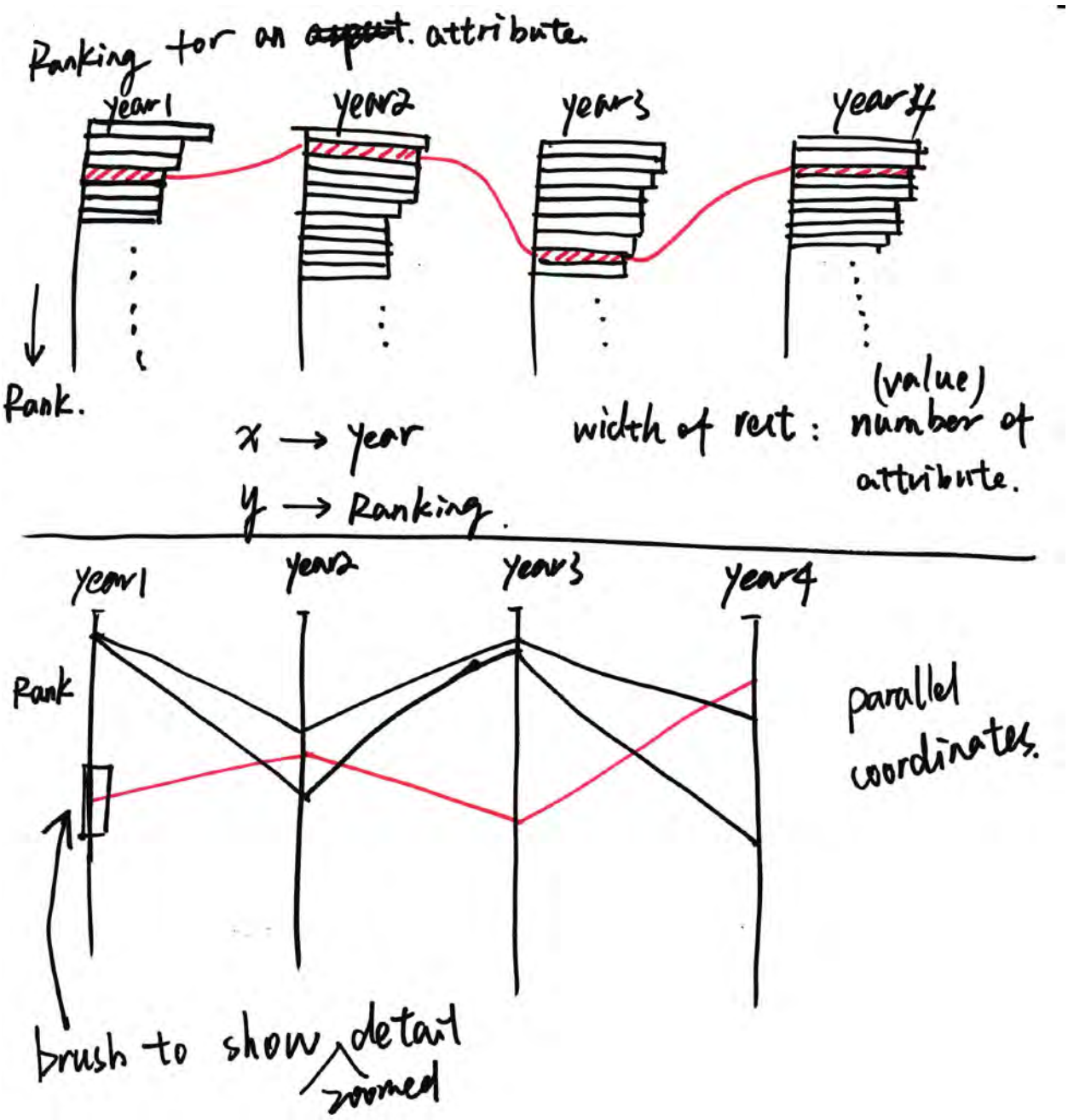

Figure 11 Designs for Ranking Charts

Moreover, for comparison purpose, we designed two views. The first one is bar chart and another one is overlap radar chart. However, overlapped chart might be difficult for users to distinguish differences. And if we want to distinguish overlapped chart, we should use different color hue. It is not good for a visualization design. Therefore, we choose to use horizontal bar chart. Users could easier to distinguish bar length to compare two players.

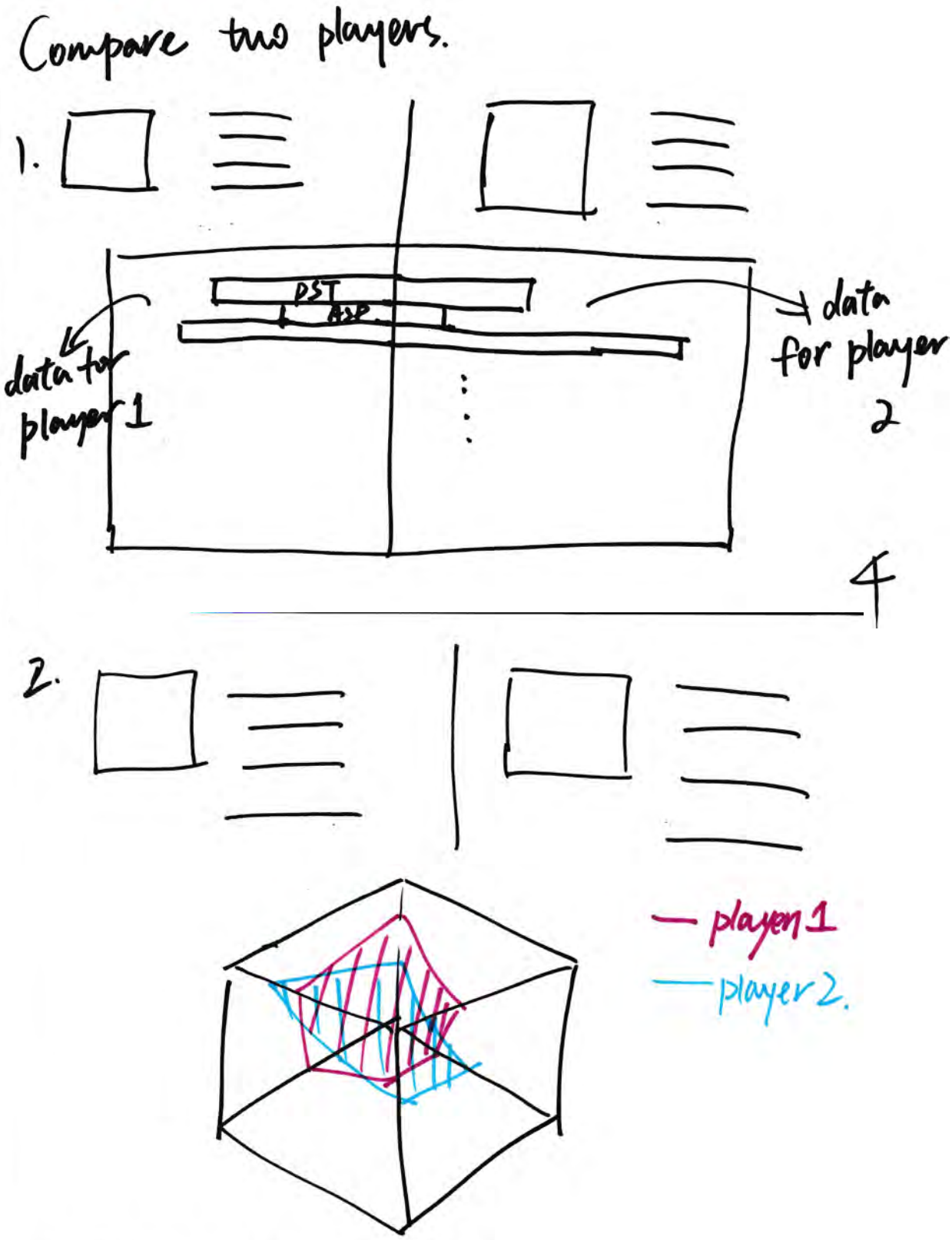

Figure 12 Designs for comparison two players

The last view design is for our optional feature, which is designed for user customizing his/her own team and compare two customized team. Users could add arbitrary players to customize a team and show the comprehensive performance of the team as figure 13 shows that.

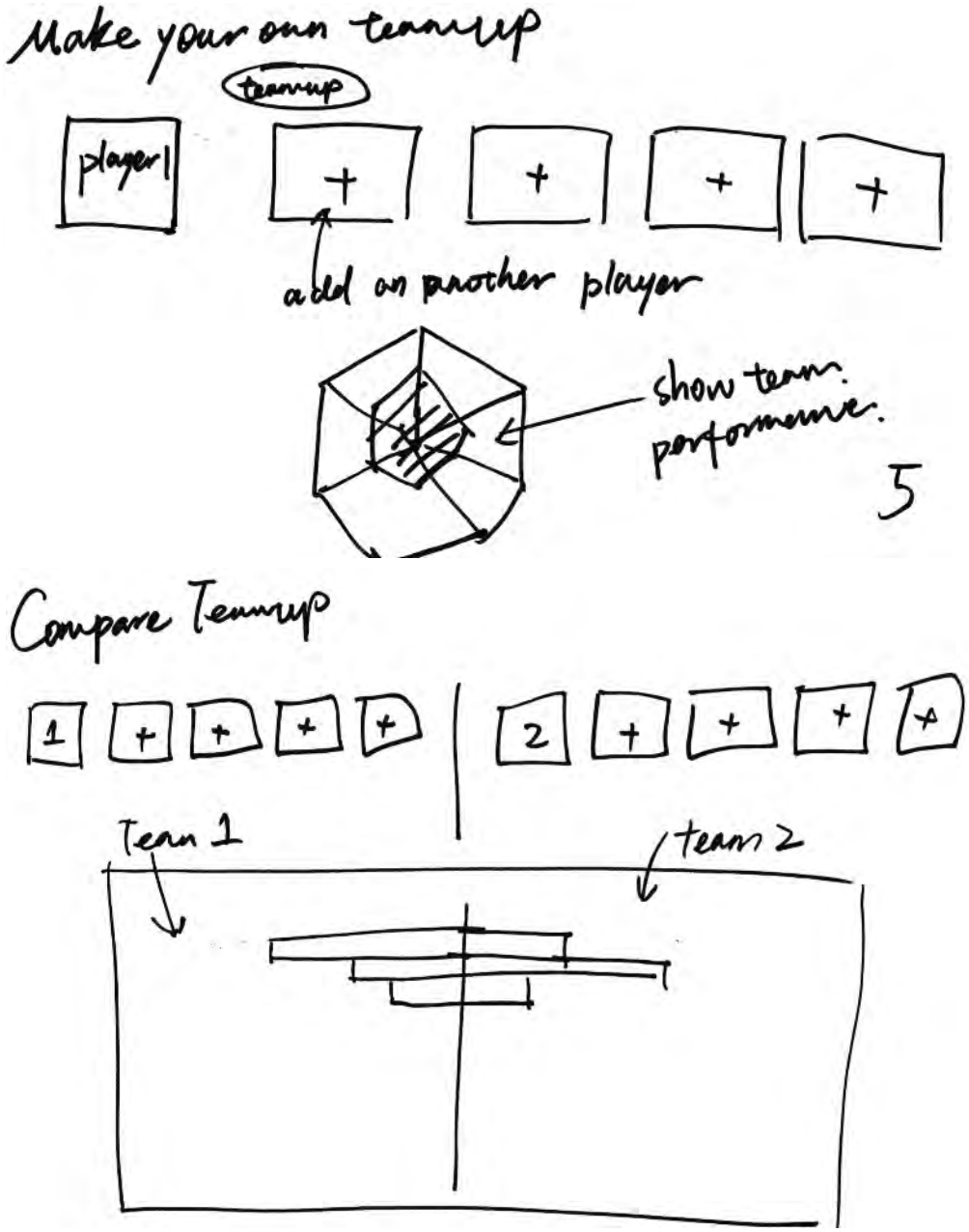

Figure 13 Designs for Customizing teams and Comparison of Customizing Team Up

Interaction Design

In our project, we design a filter on the top of our website. Users could type in players' name, players' team and position. After filtering, the filter panel would hide until users press "Change Players" button.

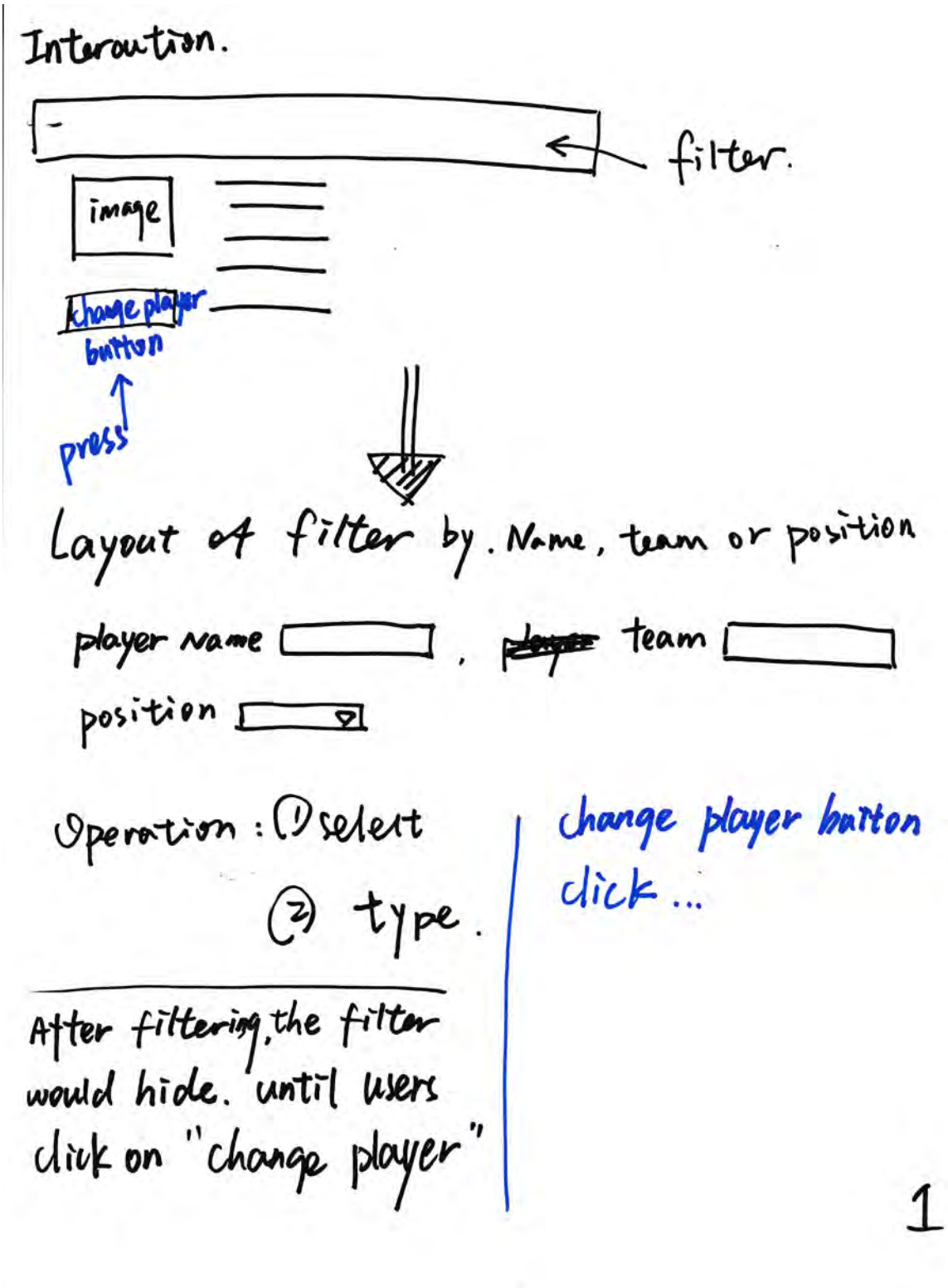

Figure 14 Interaction of Filter

For each attribute, a user could click on the one he/she wants to see detailed data. After the user clicks on an attribute, the ranking and the map of attribute's value would be displayed. Moreover, users could brush a period on year line and the ranking chart and map of attribute's value would be displayed by given years.

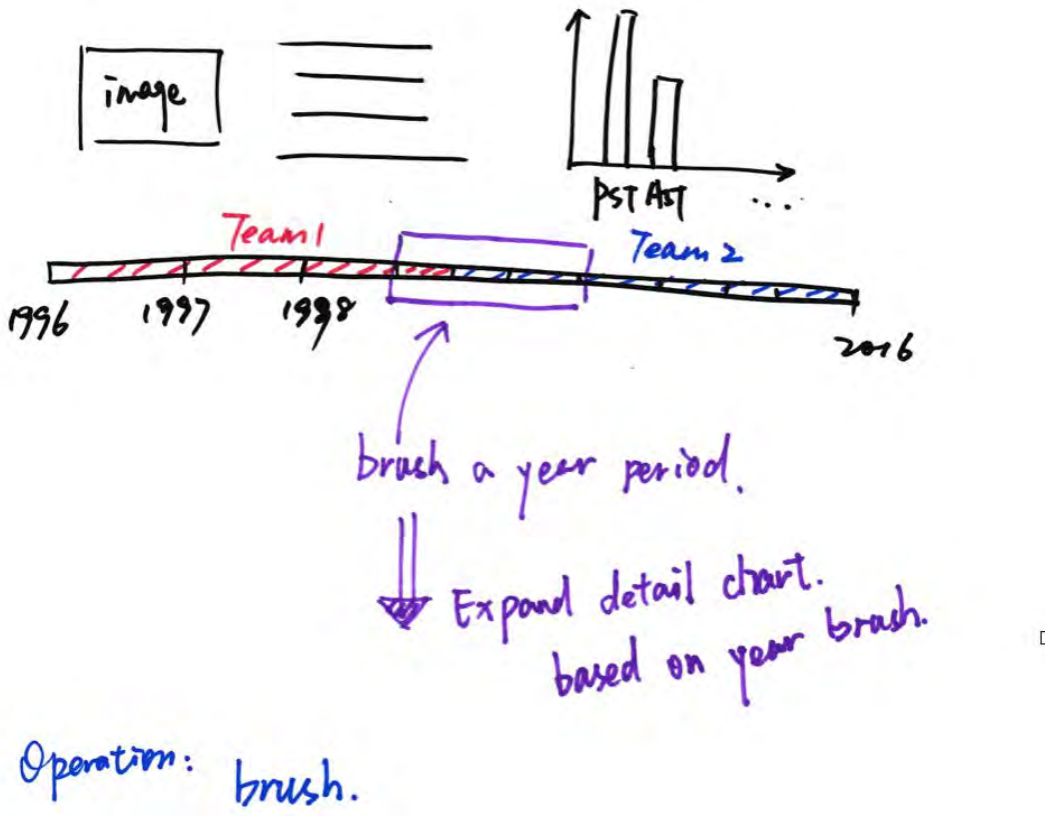

Figure 15 Interaction of choosing year period

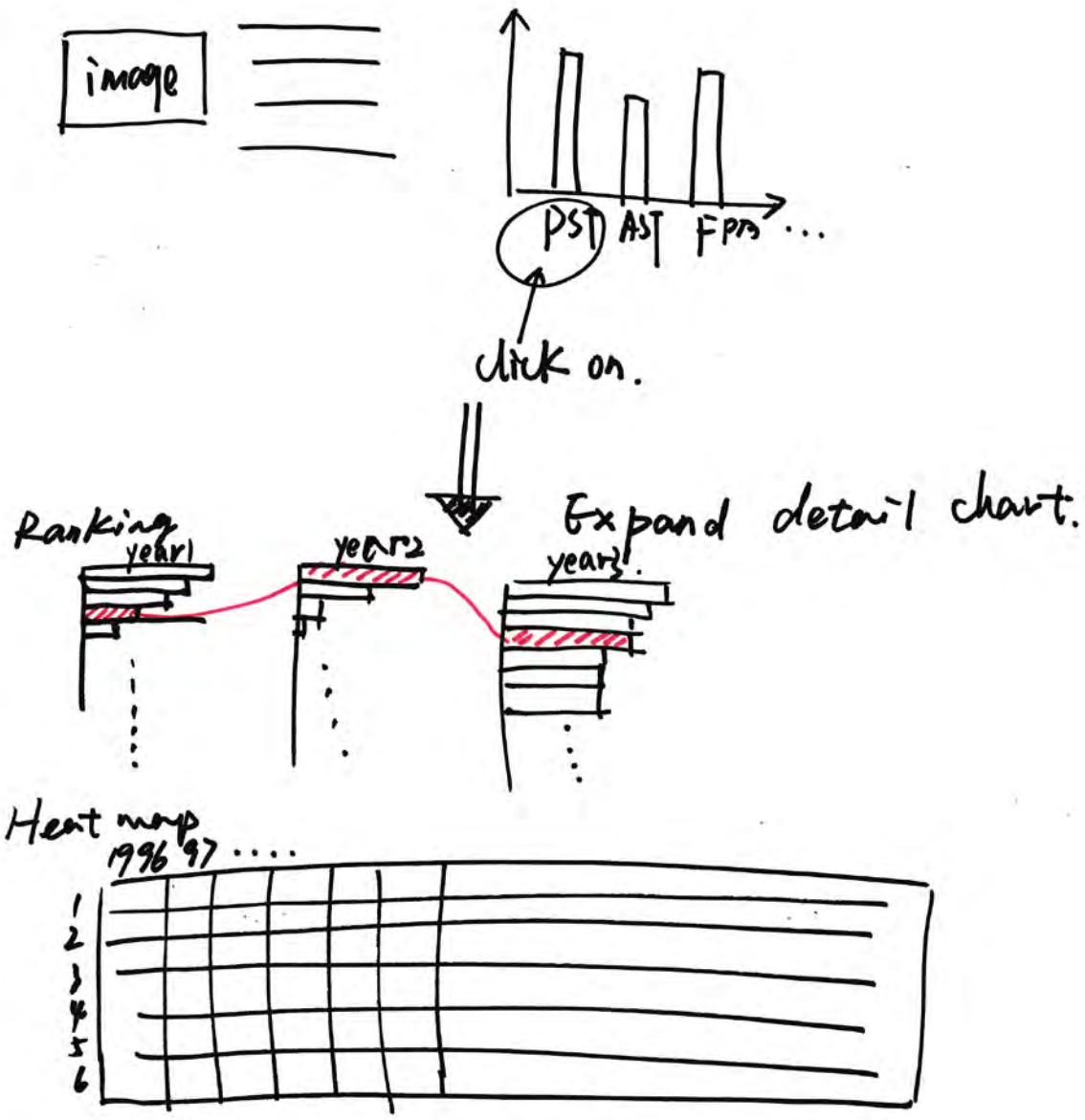

Figure 16 Interaction of displaying attribute's detail

Initial Design(Version1.0)

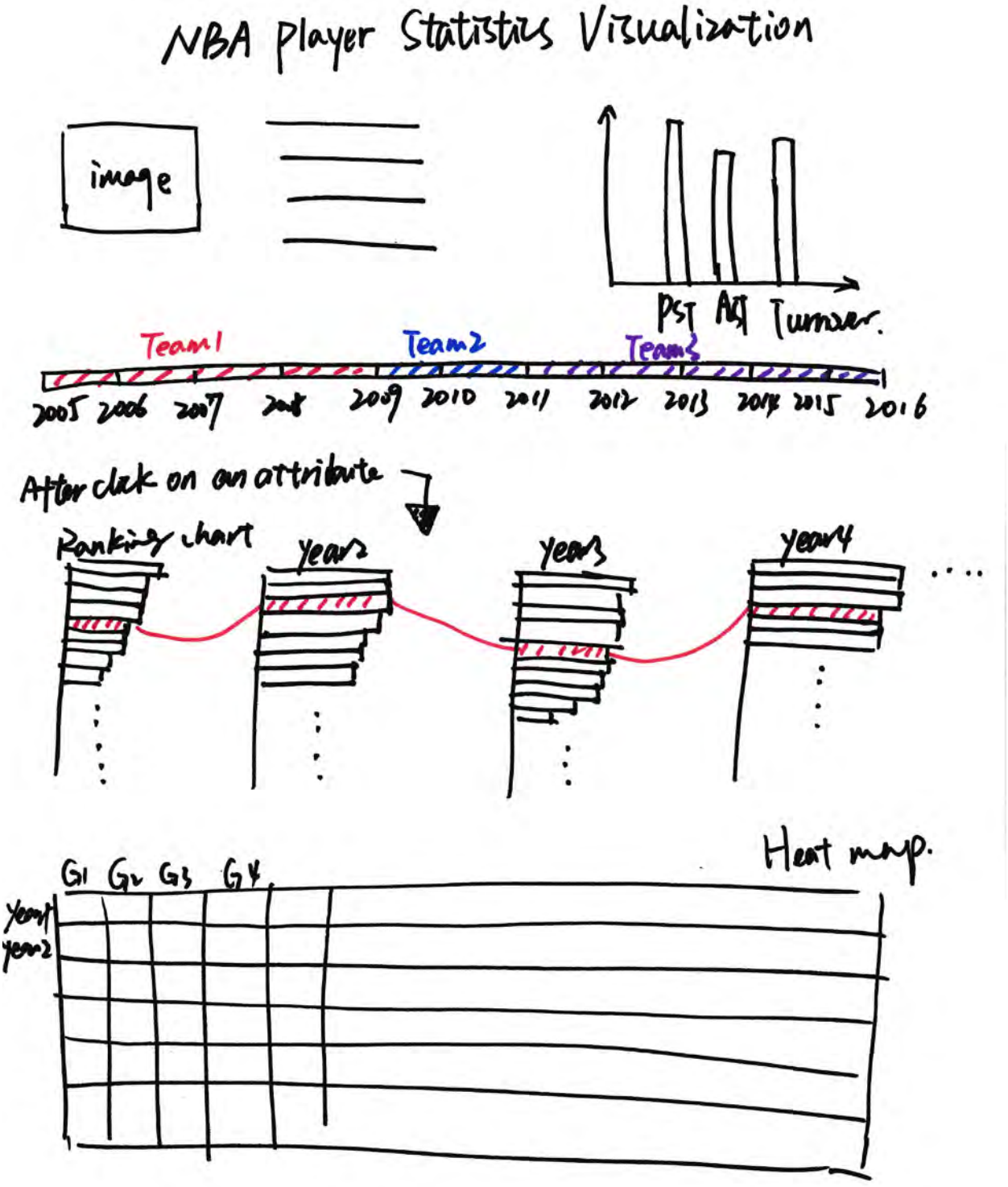

Figure 17 Design Overview

10. Implementation

So far, we implemented a prototype by using data from Kobe Bryant for our project. We display his basis information and general performance data. The attribute we used to implemented detailed charts are points. The ranking chart uses the first view we designed. Also, we have implemented the map of the attribute's value.

Figure 18 Website Prototype