



NBA Player Statistics Visualization Process Book

12.2.2016

NBA Player Statistics Visualization

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

Qi WU qw@cs.utah.edu (u1077116)
Mengjiao HAN u1078078@utah.edu (u1078078)
Qihua 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

CS6630 Final Project Process Book

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.

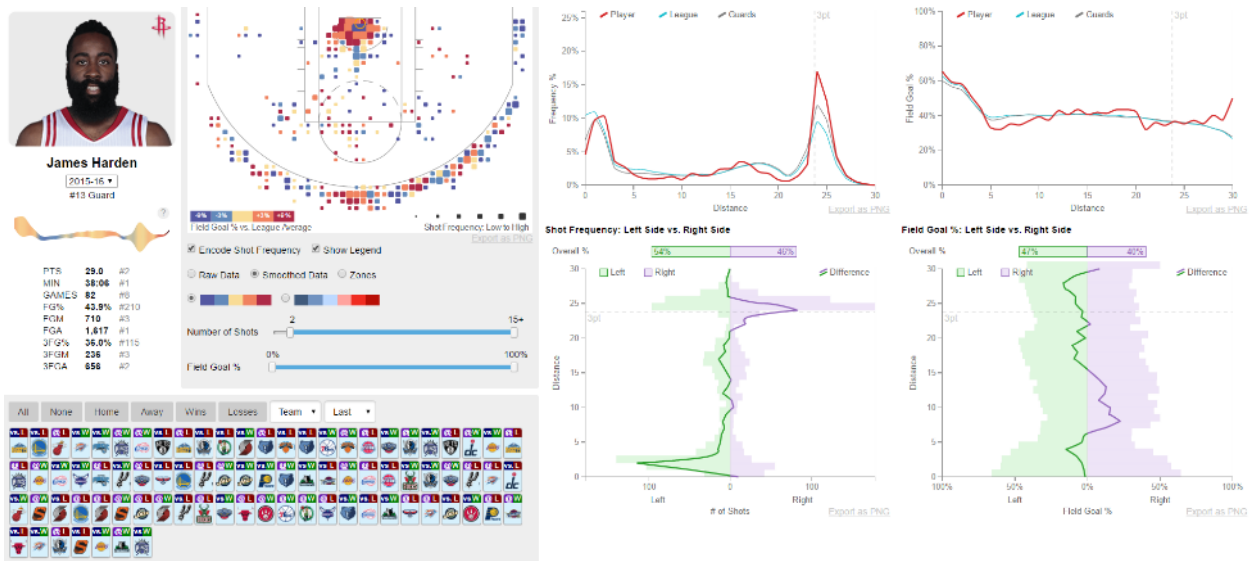


Figure 3 Peter Beshai's Visualization http://buckets.peterbeshai.com/app/#/playerView/201935_2015

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:

CS6630 Final Project Process Book

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.

CS6630 Final Project Process Book

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>).

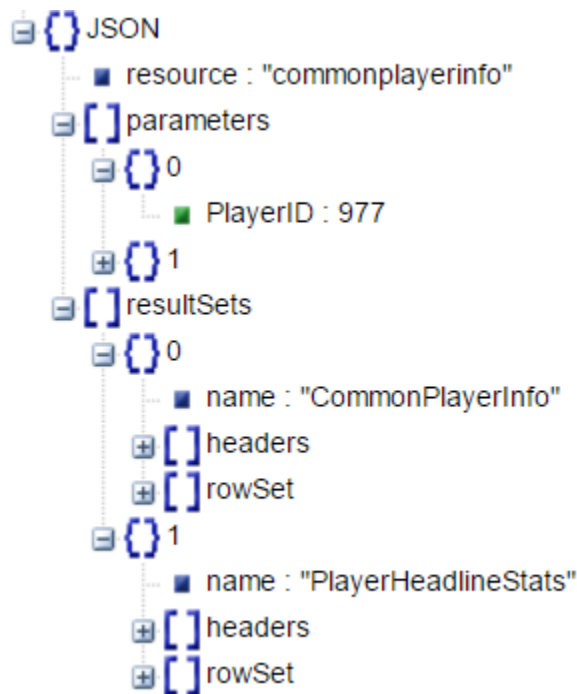


Figure 4 Raw Data Structure Generated by <http://jsonviewer.stack.hu>

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).

CS6630 Final Project Process Book

```
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 effectively 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.

CS6630 Final Project Process Book

```
— 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.

CS6630 Final Project Process Book

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 trace 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.

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.

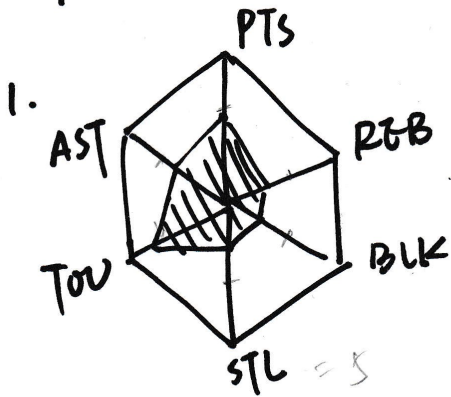
9. Design Different Views and interaction

9.1 Different views

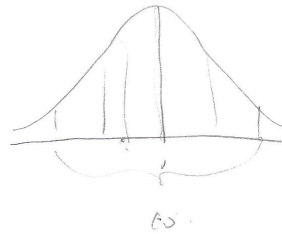
As our tasks stated above, at the beginning, we design a visualization for general players' performance. As figure 8 shows that, we design four different views to show a player's general performance. The first one is a radar chart. On radar chart, each angle represents one aspect, for example, points, 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.

CS6630 Final Project Process Book

players' performance balance.



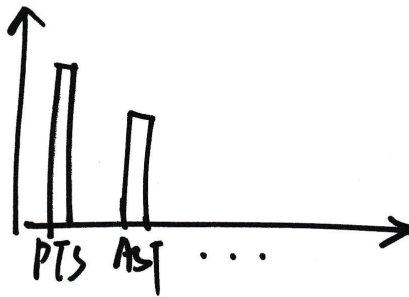
radar chart



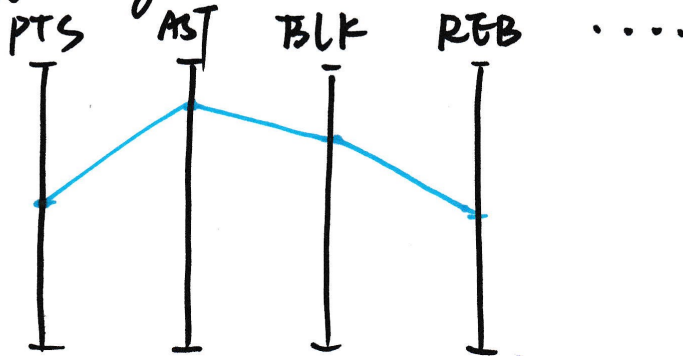
2. Table

PTS	AST	TOU		

3. bar chart



4. Ranking.



1

CS6630 Final Project Process Book

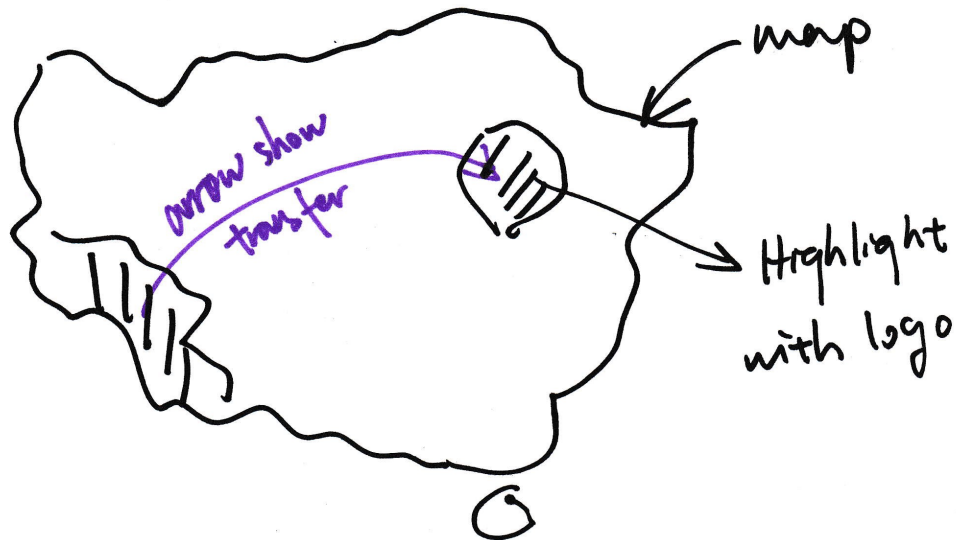
Figure 7 Designs for player's performance balance

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.

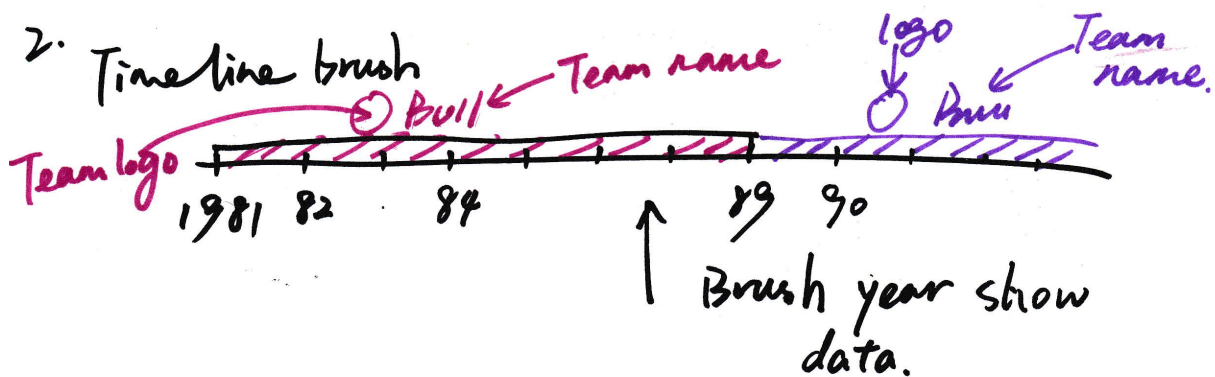
CS6630 Final Project Process Book

Team Transfer

1. map



2. Timeline brush



3. text

Bull 1981-1989

~ 1990~..

Figure 8 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

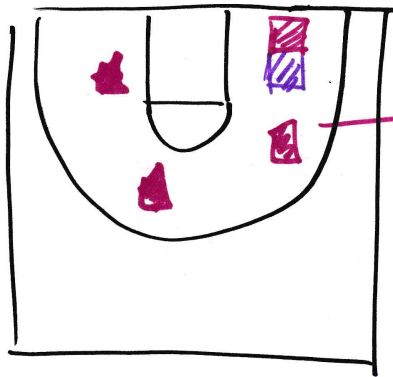
CS6630 Final Project Process Book

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.

CS6630 Final Project Process Book

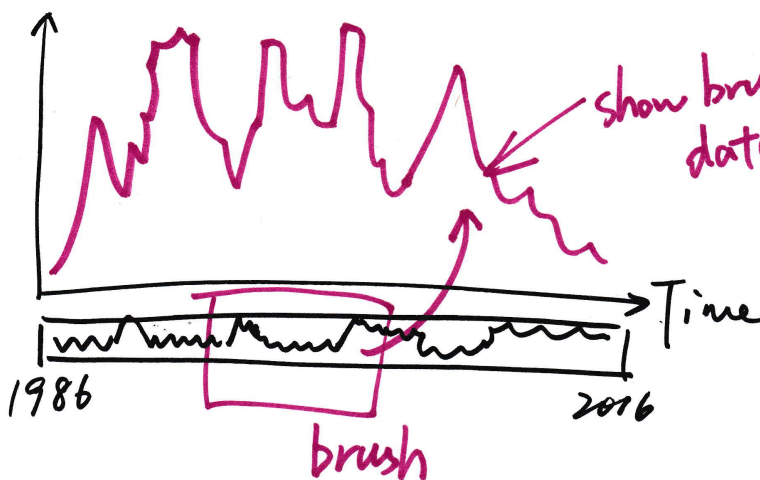
Data Detail. eg. PST, AST....

1.



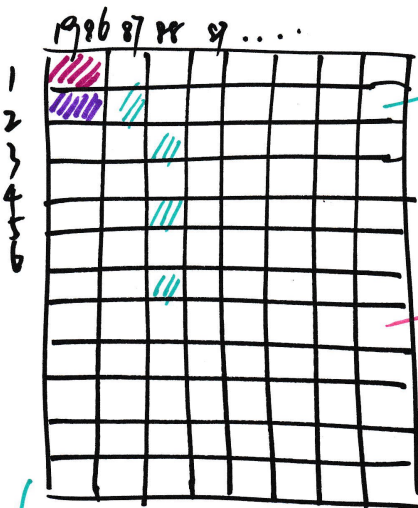
Use color value show relation between data and position.

2.



show brush year data.

3.



use color value show data, such as PST, AP....

USE Color Values Not hue. Like heat map.

game

page 10

CS6630 Final Project Process Book

Figure 9 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.

CS6630 Final Project Process Book

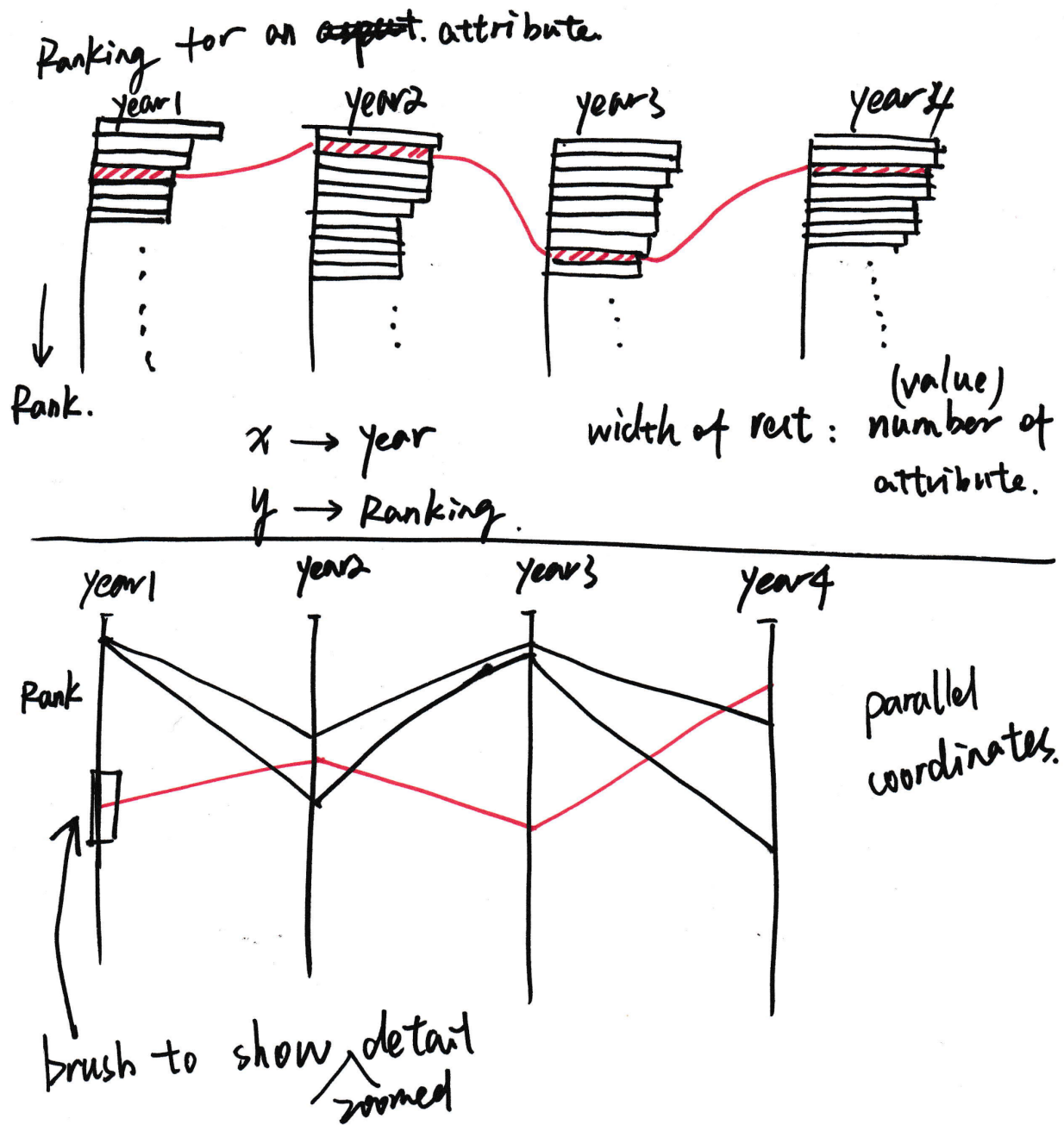
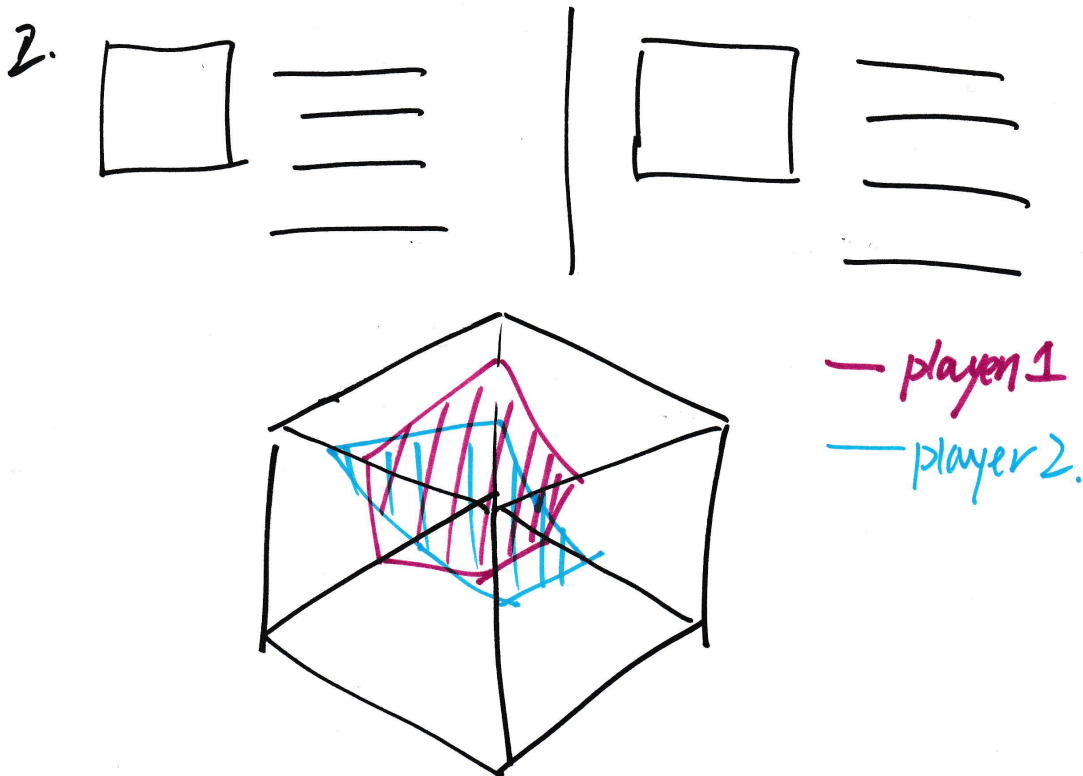
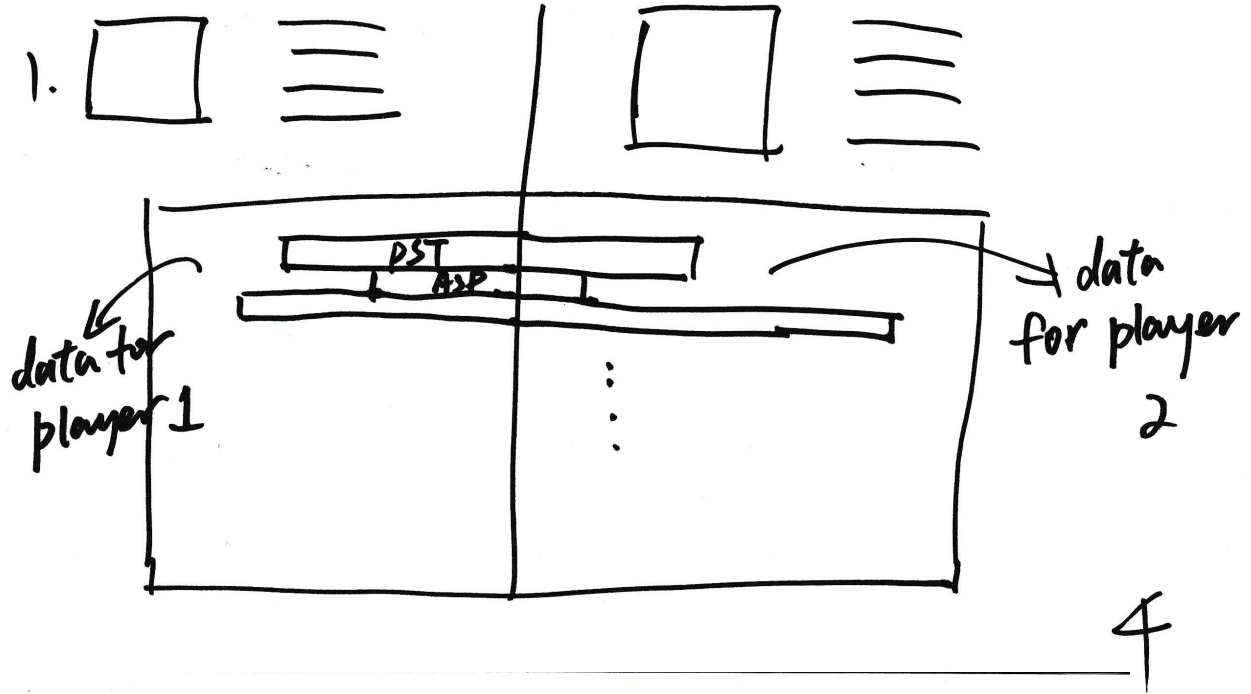


Figure 10 Designs for Ranking Charts

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.

CS6630 Final Project Process Book

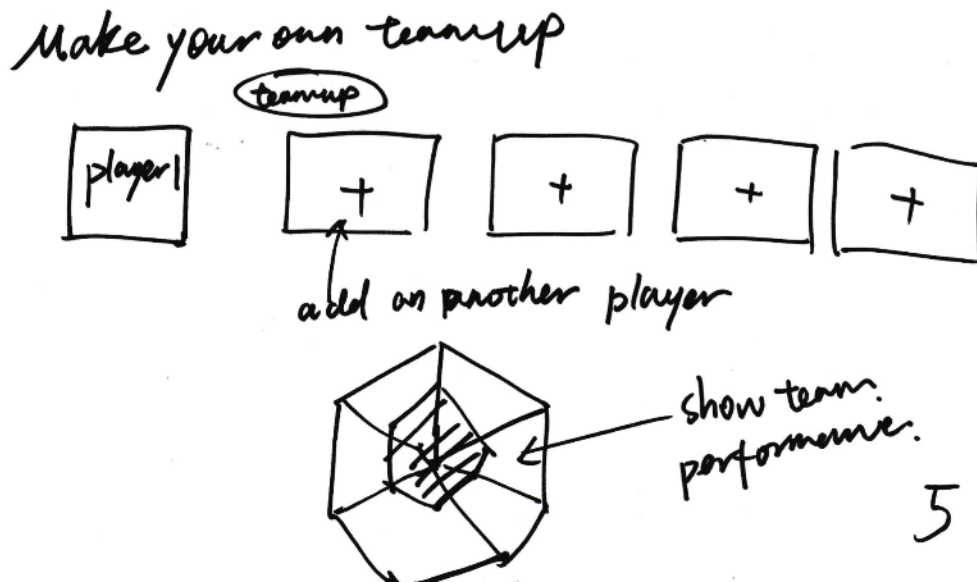
Compare two players.



CS6630 Final Project Process Book

Figure 11 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.



Compare Teamup

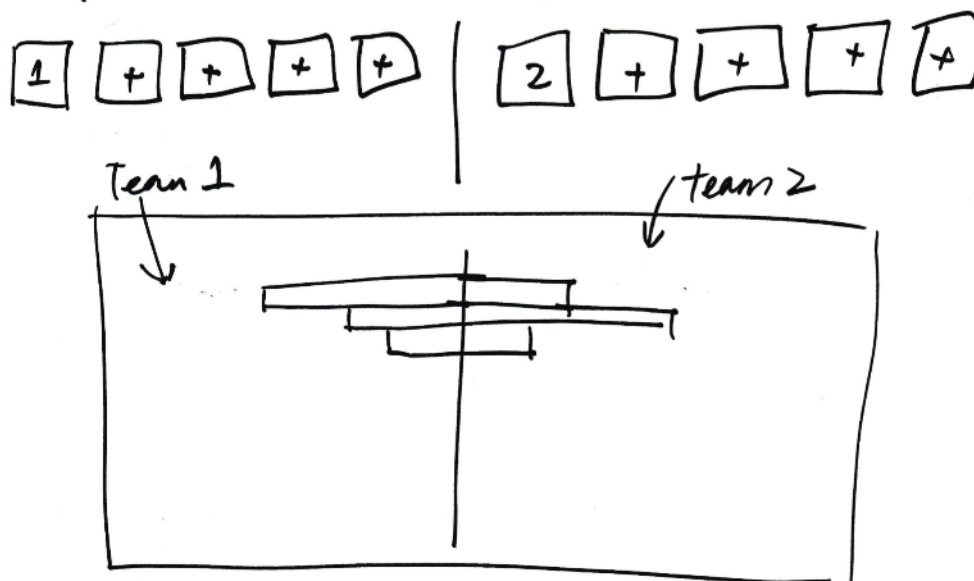


Figure 12 Designs for Customizing teams and Comparison of Customizing Team Up

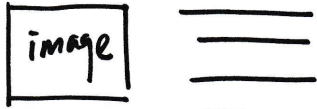
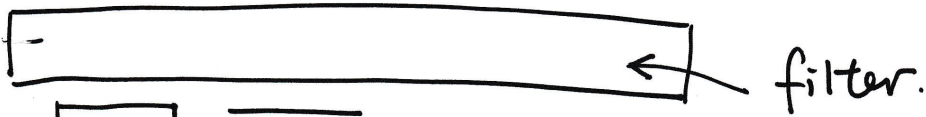
CS6630 Final Project Process Book

9.2 Initial 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.

CS6630 Final Project Process Book

Interaction.

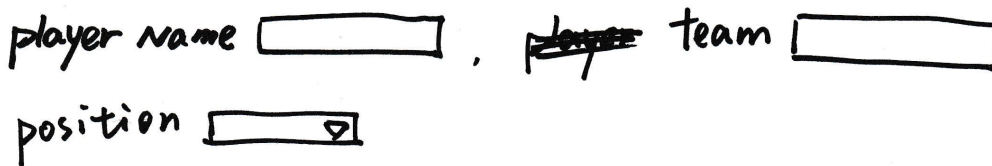


change player
button

press ↑



Layout of filter by. Name, team or position



Operation: (1) select
(2) type.

change player button
click ...

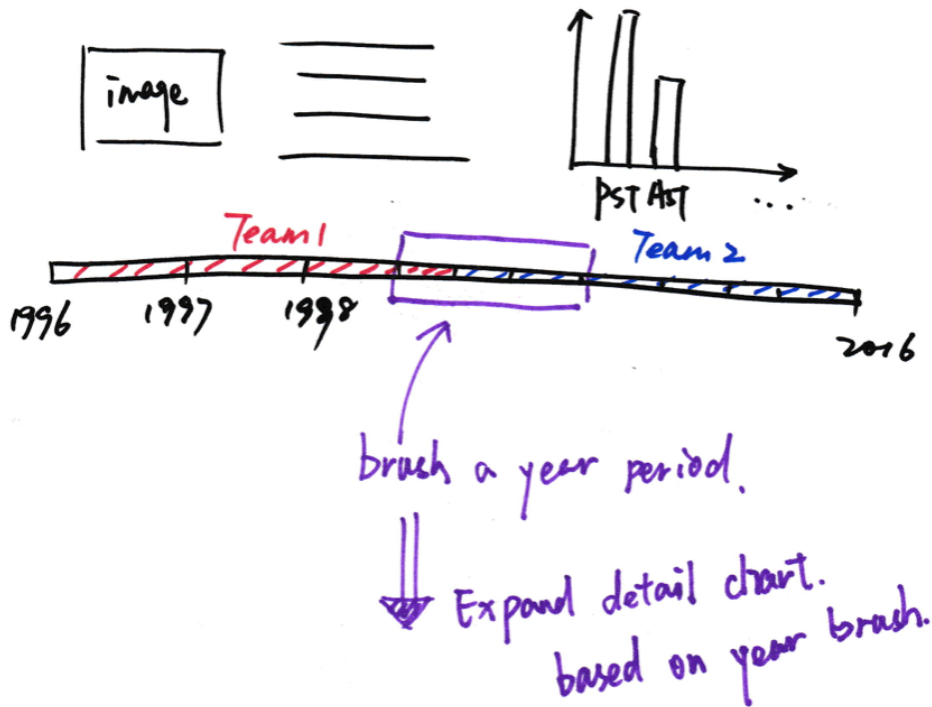
After filtering, the filter would hide. until users click on "change player"

1

Figure 13 Interaction of Filter

CS6630 Final Project Process Book

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.



Operation: brush.

Figure 14 Interaction of choosing year period

CS6630 Final Project Process Book

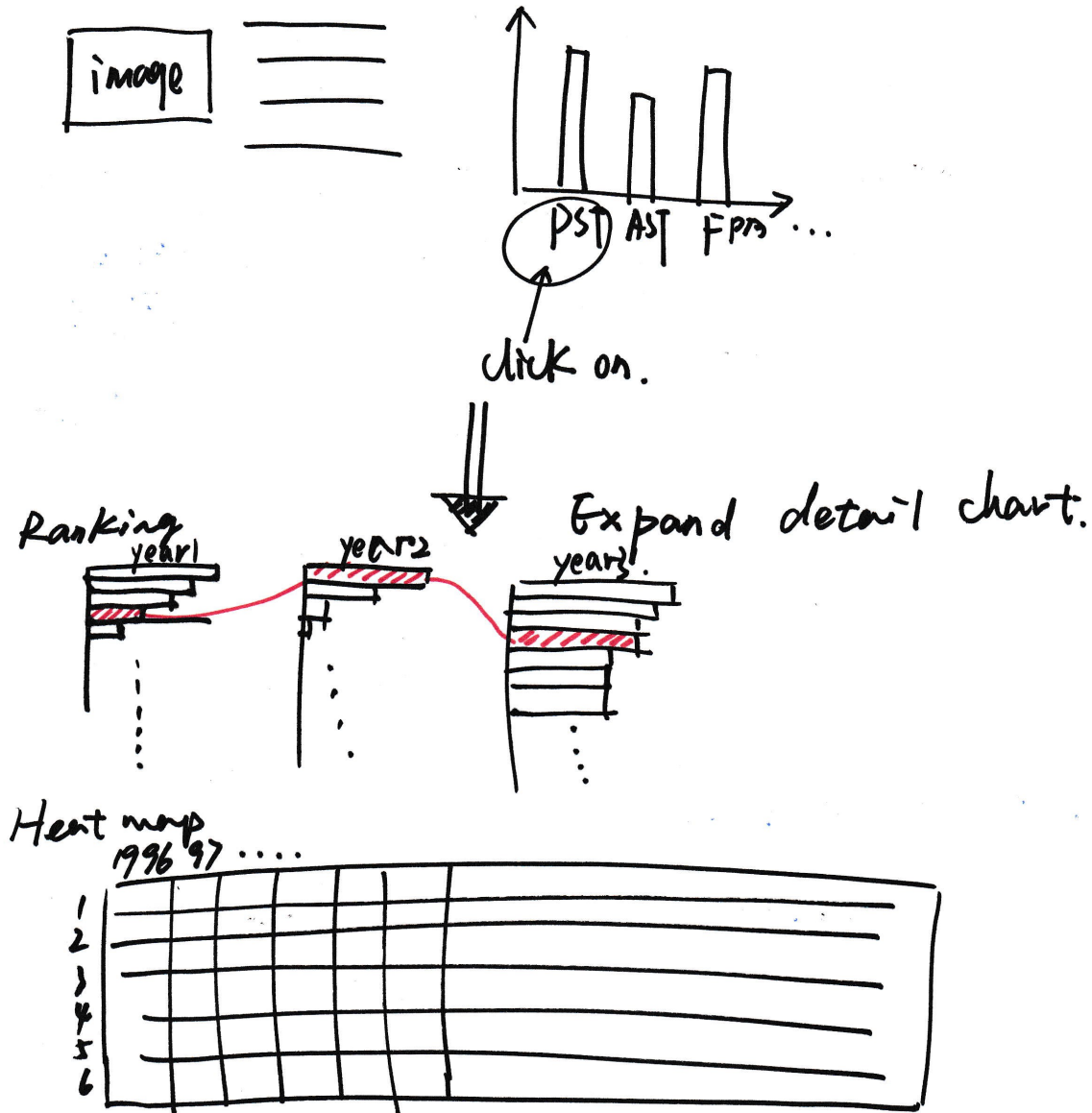


Figure 15 Interaction of displaying attribute's detail

CS6630 Final Project Process Book

9.3 Initial Design(Version1.0)

NBA Player Statistics Visualization

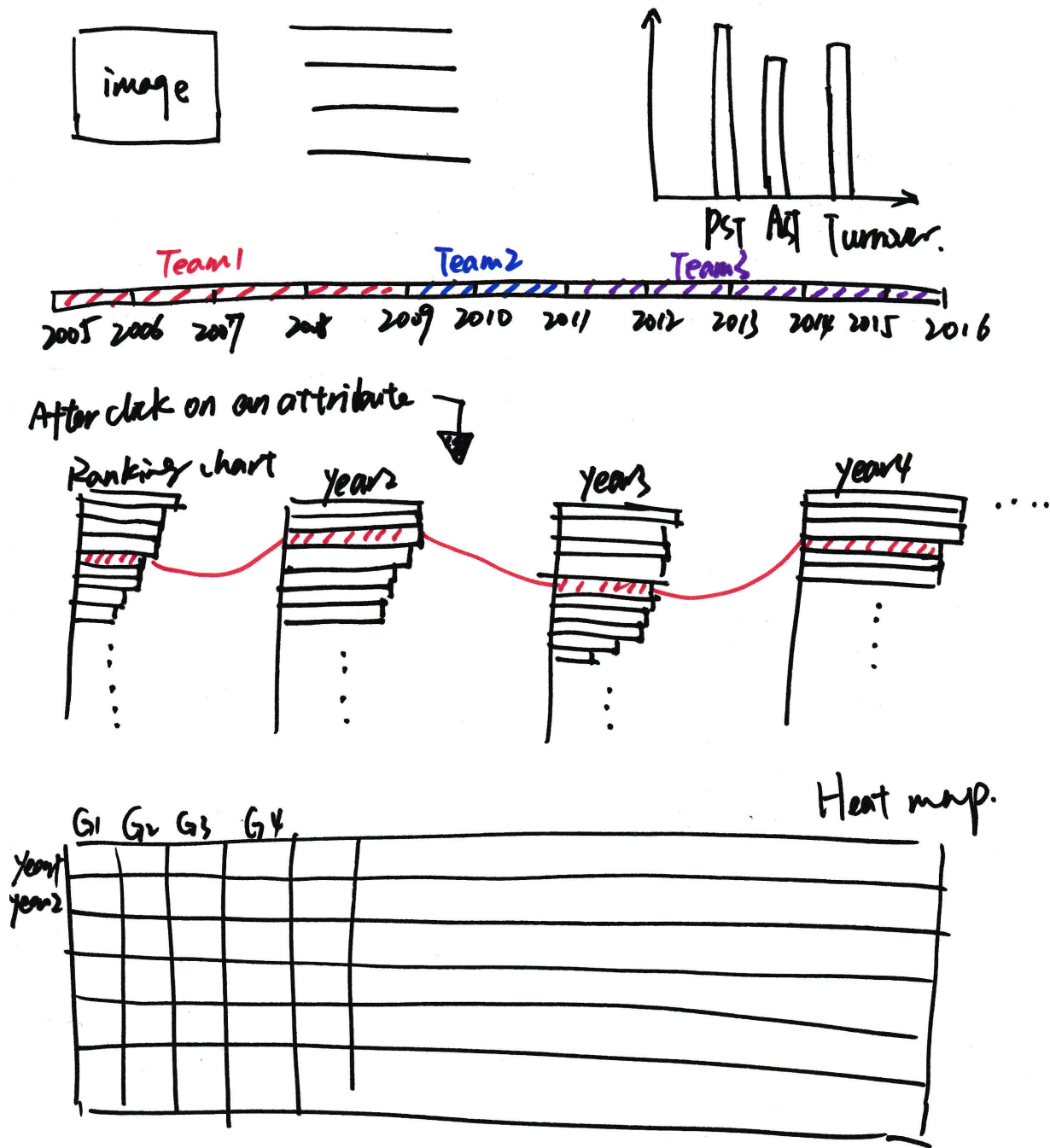


Figure 16 Design Overview

CS6630 Final Project Process Book

10 Design revolution

Based on our initial design view as figure 17, firstly we tried to code each view separately. During this process, we found more information and our data and we also change design view to make sure all designs are reasonable and all data are encoded correctly. The following section is our design revolution based on the initial design view.

10.1 Filter

Design of filter is as **figure 18**. Our design of filter is simple. Users could select team name or position to query players. Or, users could type the name of players. At first, we just use a whole list of player name as figure 19. It is definitely not an effective design, what we were doing is to test select one player and display his page.

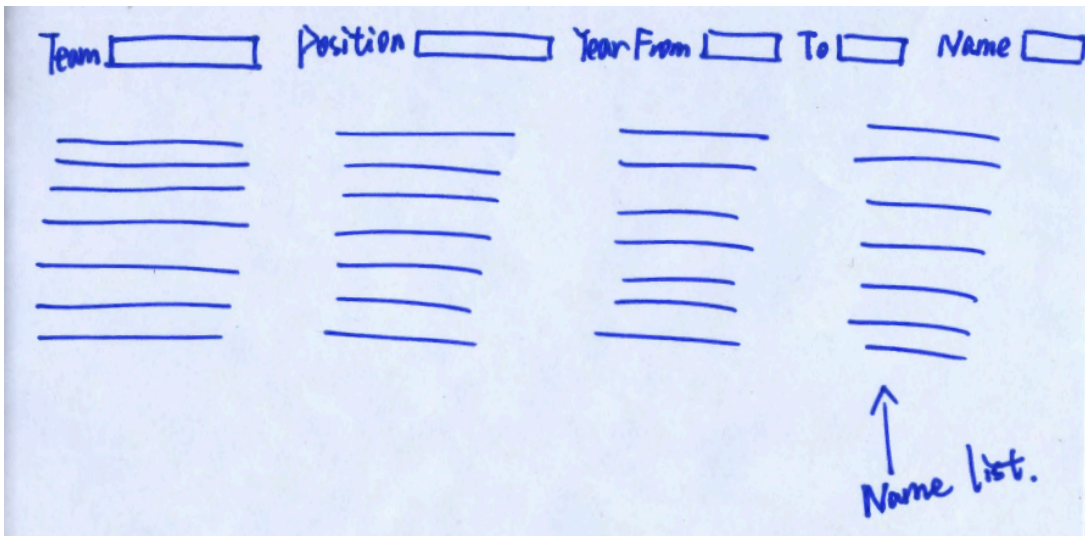


Figure 18 Initial design of filter

CS6630 Final Project Process Book

10.2 Basic information View

The basic information of players is simple, which includes player's image and some basic information. What we were focus on is the chart on the right of basic information view, which shows the players overall performance in six important skills. Our initial thought is using radar chart as figure 22. Radar charts are primarily suited for strikingly showing different kind of player's performance under multidimensional view and it could implement for ordinal measurement. However, it is hard to encode data because of different scale for each attribute. And, it is hard to visually compare lengths of different performance, because radial distances are hard to judge, though concentric circles help as grid lines. Future, the connection between each attribute does not make sense. It would lead users to compare them. Instead, we use a simple bar chart graph at first as figure 23 shows that.



Figure 22 Radar chart design

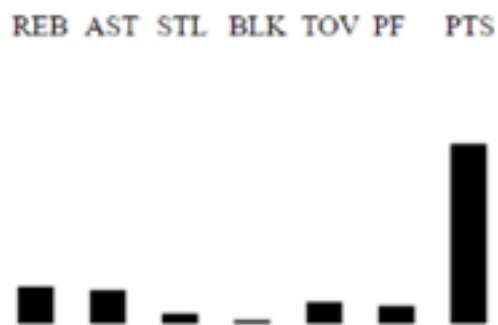


Figure 23 Bar chart for showing overall performance

CS6630 Final Project Process Book

Although bar chart is simple, it may also mislead users to compare among skills and scaling problem still exist. Therefore, we accepted Sean's suggestion to use a radial column chart as he showed in class, which deletes the connection between each attribute. Moreover, we defined our method to scale. For each attribute, we use ranking from recent year as a benchmark. Then data of ranking twenty would be mapped to 1, and others data below would be mapped to 0 and 1 uniformly. All data above is mapped as 1.

Figure 24 shows the design result. And we added an Offense-defense balance bar, using it to show what kind of skill that a player is more good at. For example, if the result near offensive, the player would be an offensive player.

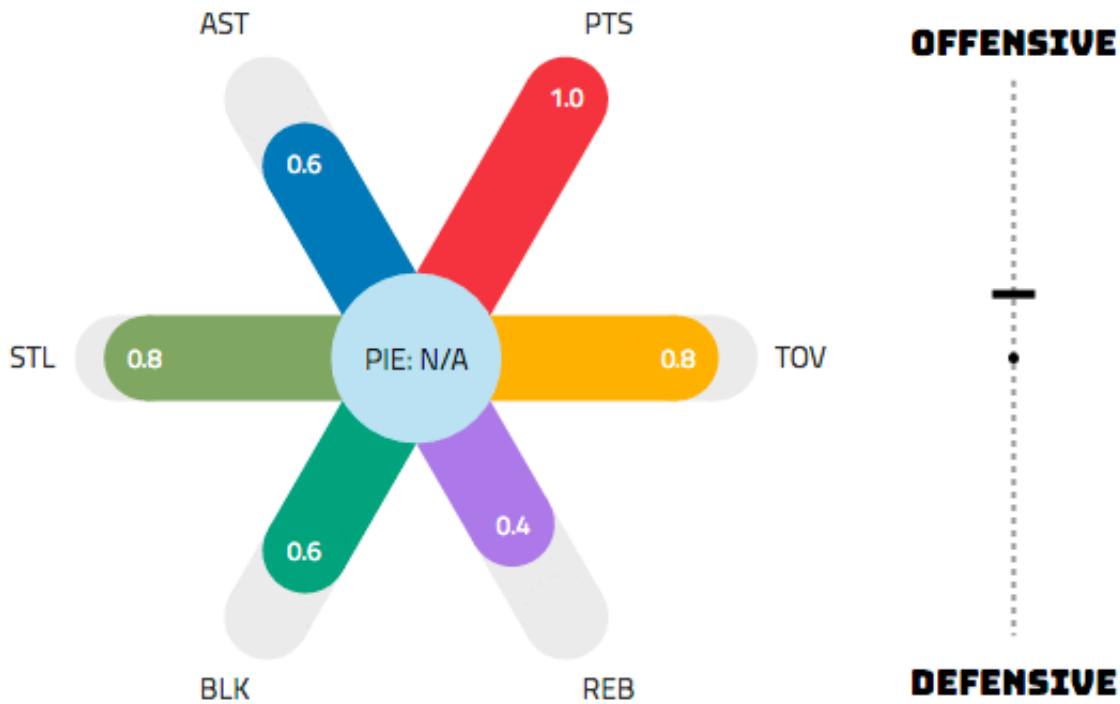


Figure 25 Final design view of overall performance

CS6630 Final Project Process Book

For interaction design, we added tooltip for each bar in our radial column chart to display the real number of data as figure 26 shows that.

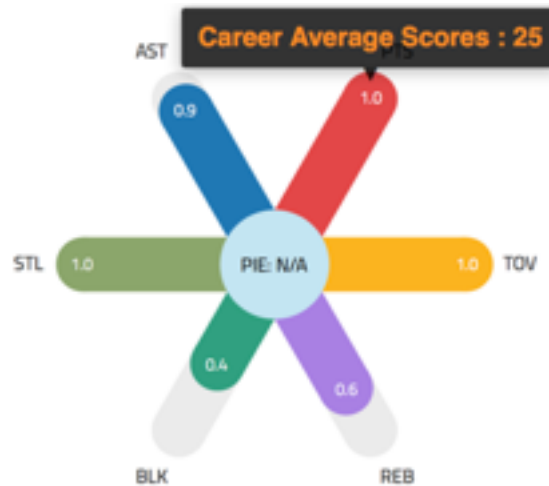


Figure 26. Implementation of radial column chart

Moreover, we added a tooltip to explain the meaning of each attribute and the scale for each attribute as figure 27 shows that. And click on each text of attribute would change the display of ranking view and detailed data view, which would be described below.

CS6630 Final Project Process Book

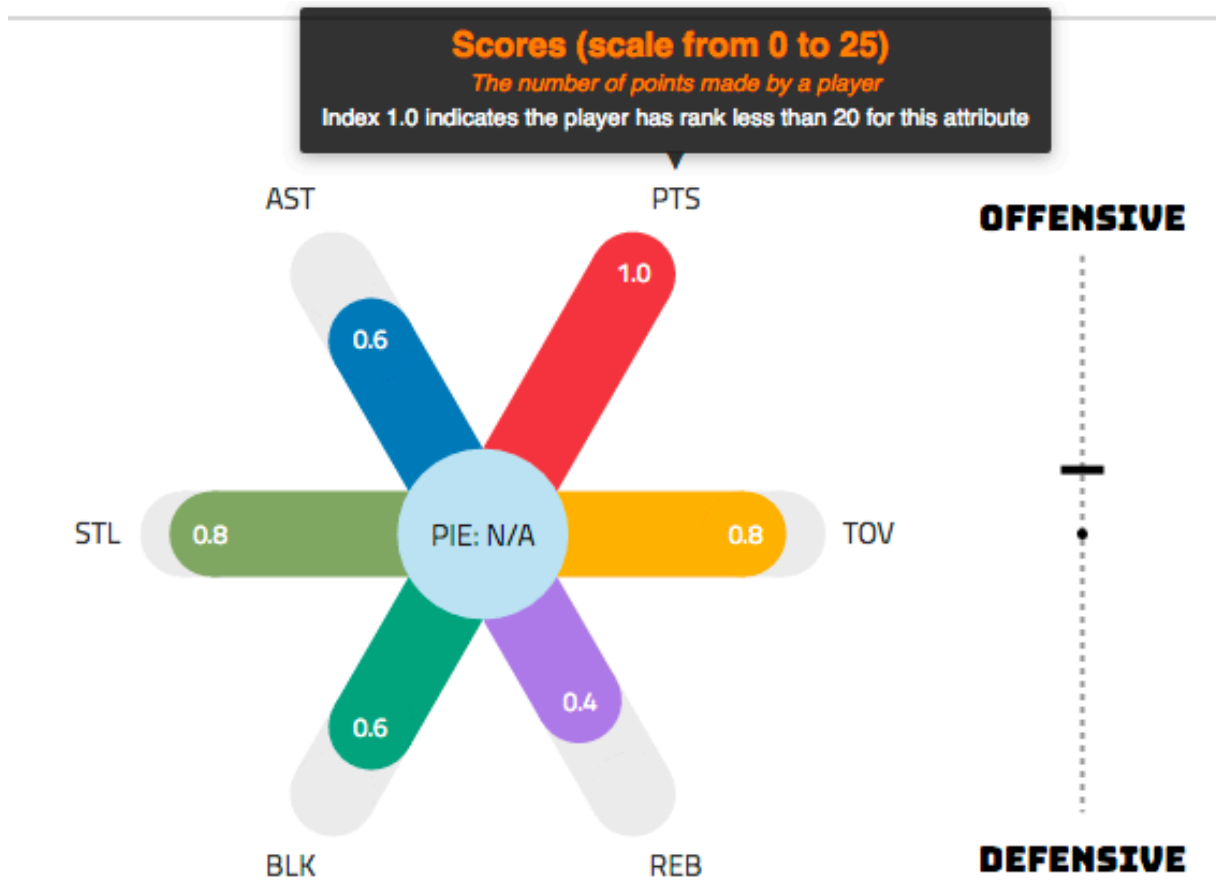


Figure 27 Implementation of radial column chart

Another part of basic information view is year time line. We decided to use time line to present the team transfer. So team logos are appended into time line. And we use the team's preference color for each time interval. Figure 28 is the final design of basic information view.

CS6630 Final Project Process Book

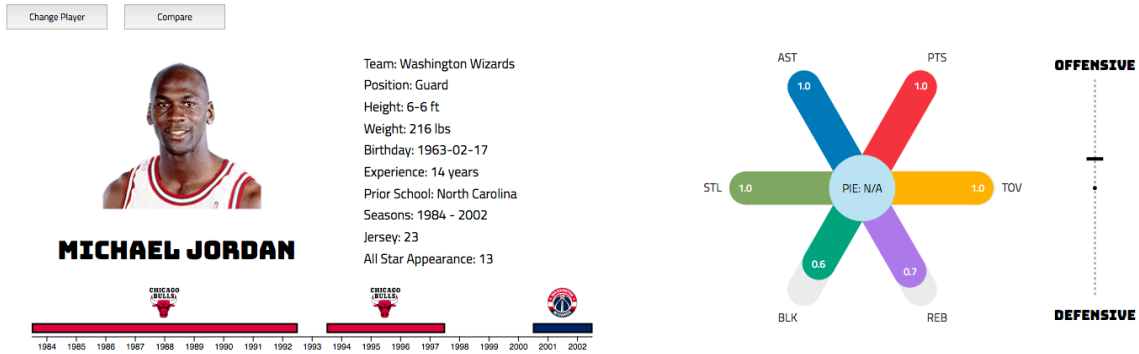


Figure 28 Final design of basic information view

10.3 Ranking View

At first, we tried to use our initial design as figure 29 shows that. However, after first implementation we decided not to use rectangular bar, because there are a lot of players in each year around 200 to 300, so it makes the whole chart really long and big. Therefore, we set the height of each rectangular to 1, it looks like a lot of lines as figure 30 shows that.

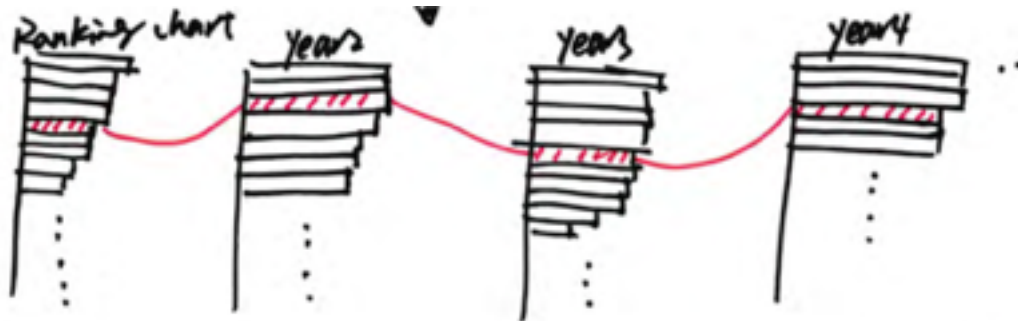


Figure 29 First design of ranking chart

CS6630 Final Project Process Book

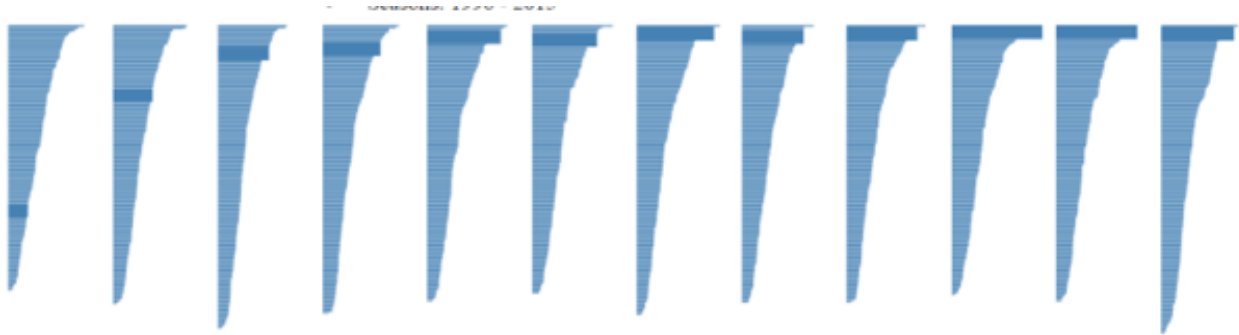


Figure 30 First change of ranking chart

The view in figure 30 is not effective. It is messy and because we use a really small height of rectangular, they may overlay sometimes that we do not know reason. So, we tried to avoid the width of each line, which represents the real number of data for one player. And then we designed view as figure 31. In figure 31, just player's ranking in every year is showed.

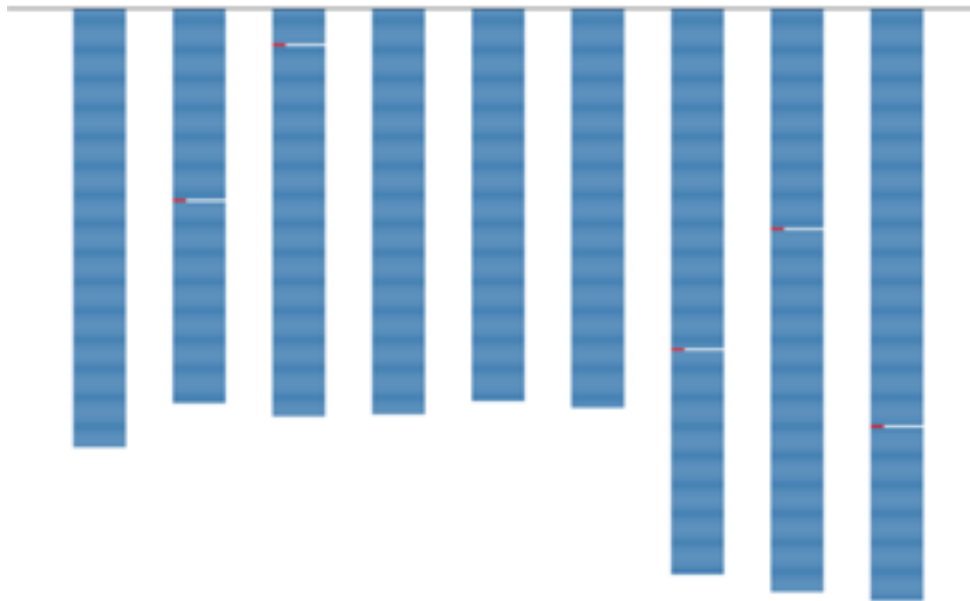


Figure 31 Second change of ranking view

Compared figure 31 with 30, it cannot provide information of data for each player in each year. Therefore, we prefer to improve the view of figure 30 instead of avoiding the real data for each player. Moreover, drawing that many bars is really slow. To improve that view, instead of using bars with small height, we just use a line to indicate attributes value boards are added to make sure the view is not so

CS6630 Final Project Process Book

messy, and a little circle is added when highlight the current player's ranking position as figure 32 shows that.

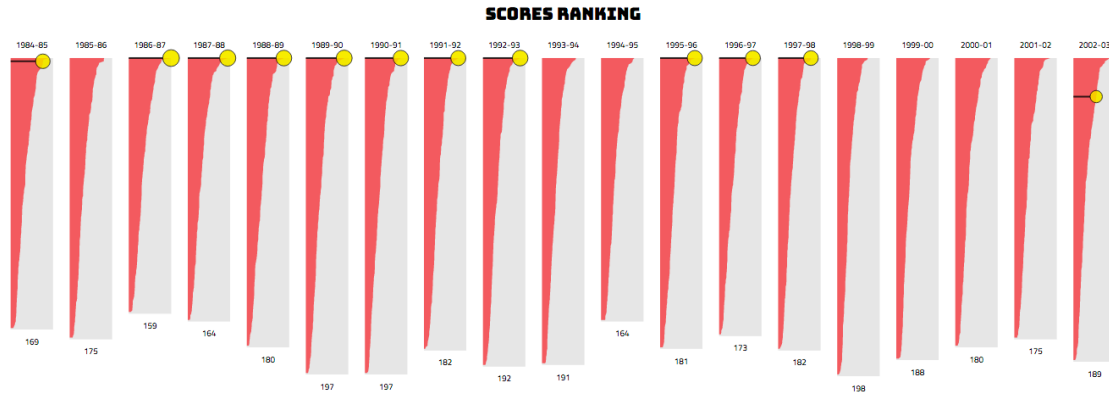


Figure 32 Final design of ranking view

After that we connect each attribute in overall performance radial column chart, and they are consistent by using same color.

For interaction, because our task is to let users know the ranking of selected player and also they could know whose rank is around that player. Hence, we added a tooltip as figure 33 shows. When users hover the mouse over.

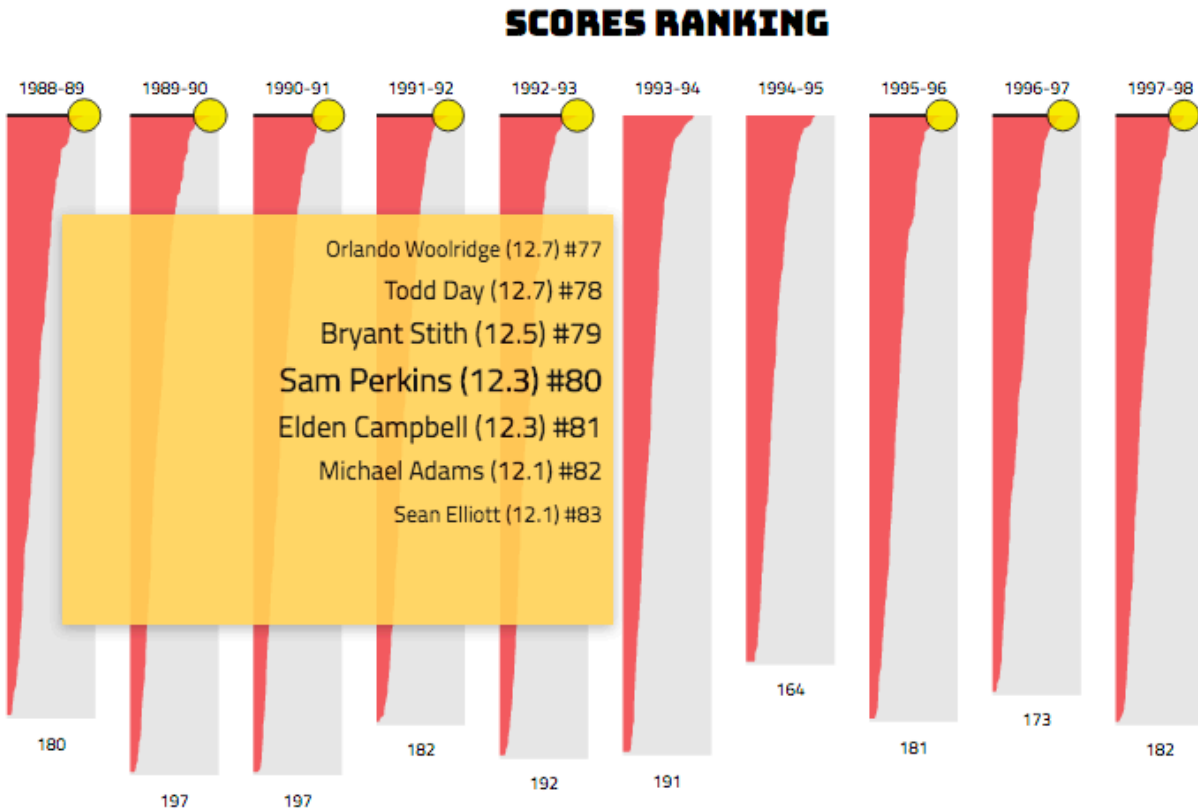


Figure 33 Interaction of ranking view

10.4 Attribute Detailed Data View

In this part, we want to further explain each player's detailed attributes according to user's selection. In general, we want to exhibit the player's performance as time goes by, using three methods to visualize. First, we displayed the player's attribute in terms of a 2D histogram, with proper binning. The X-Axis indicates the month and date when the game was played while the Y-Axis indicates the year value. It shows the temporal correlation of a player's performance on each attribute. The histogram was visualized by a matrix, with average attribute value encoded by color. Second, we used two 1D histograms to show the player's averaged performance change in terms of different periods of a season and different years. The bar chart on the right shows the histogram binned by year value, which elaborates the player's average performance for each season. The histogram on the bottom shows the player's average performance during one single season as time goes by. We can easily figure out a player's performance in different period of a season with this figure.

Therefore, our initial design is shown blow.

CS6630 Final Project Process Book

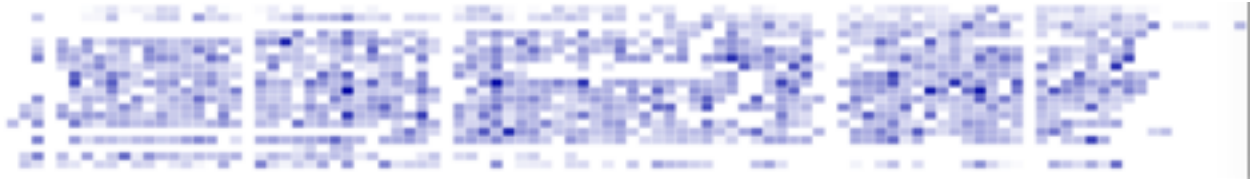


Figure 34 Initial design of detailed data view

This is our basic design chart with each bin represent intensity of the performance attributes. This 2D histogram counts the occurrence of combinations of range performance with time.

In order successfully overlay a time dimension with player's performance attributes over a 2-dimensional plot, it is essential that the time axis of the data passed to the shade in units of performance attribute.

This time, we added year as y-axis and month as x-axis, the design is shown in the picture below:

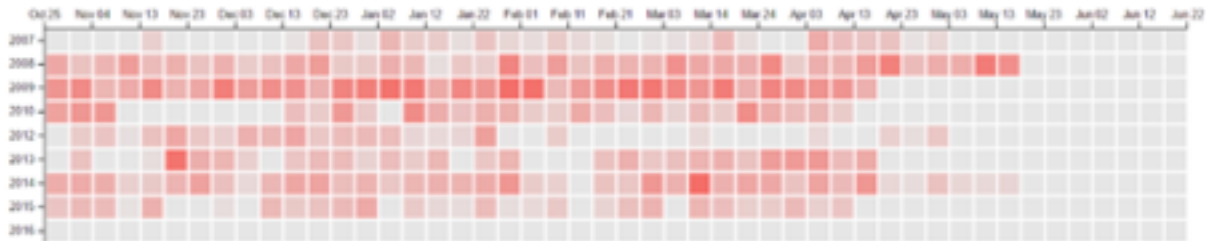


Figure 35 Change design of detailed view

However, this is only the part of the information displayed, we also need to show the player's averaged performance change in terms of different periods of a season and different years. Therefore, we changed our design by adding average performance data in each season with horizontal bars on the right side and averaged performance change at bottom.

The updated graph is designed like this:

CS6630 Final Project Process Book



Figure 36 Revolution of detailed data view

Now, we are successfully display the 2D histogram. However, in order to create harmonious color effect with the whole page. We further improve the chart by changing the color and hue. In addition, we added title on the top of the graph and a small coordinate on the lower right corner to make sure that user can understand the statistics value of average performance between each season and different years.

In the final process, we also add interaction that show the average score when user's mouse passes the bin, which can support the user in focusing on value of an exact bin.

So the final design is show the picture below:

CS6630 Final Project Process Book

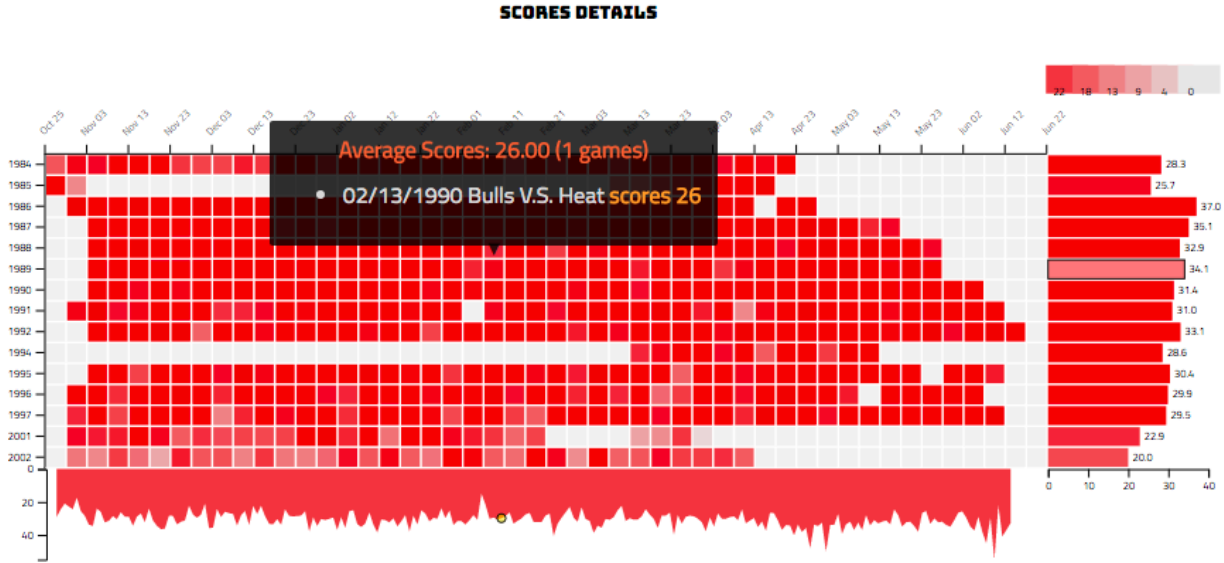


Figure 37 Final design of detailed data view

10.5 Shoot Frequency View

View of shoot frequency chart does not change a lot. However, we did not consider the method of encoding data. Data of shoot chart contains x and y position, and points got in this position. We calculated histogram from raw data.

Firstly, we tried to treat each point position as a signal pixel. The size of chart SVG component is 600 * 600 pixels, therefore, it is really slow to display the chart. Hence, we tried to discretize the chart to several pixel blocks, like several small rectangular. But fortunately, we found a library from D3, d3.hexbin (<https://github.com/d3/d3-plugins/tree/master/hexbin>). It implements hexagonal binning, which is useful to aggregating data into coarse representation suitable for display[1]. We used this plugin to encoding area. Our result view shows in figure 38.

CS6630 Final Project Process Book

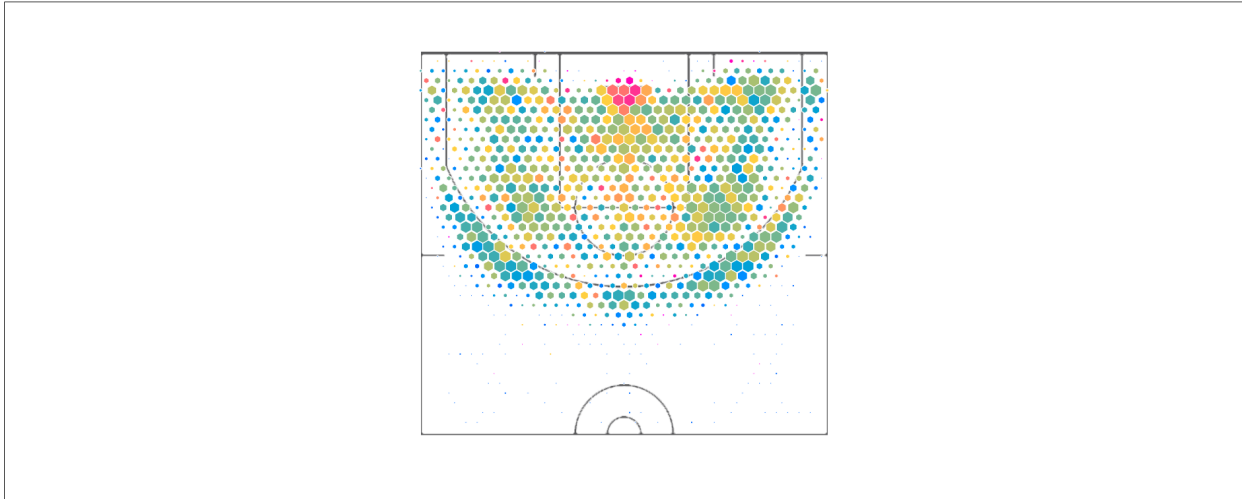


Figure 37 Initial design of shoot frequency view

Another problem is choosing discriminable and preferable color map. We prefer the data values above and below the middle respectively mapped the same color hue with different saturation. As a result, users could easily separately in which area the players' performance is better. After taking with Sean, we use a specific color palette from the website tool Color Brewer. Colors are encoded with the value of field goal percentage. Our result is in figure 39.

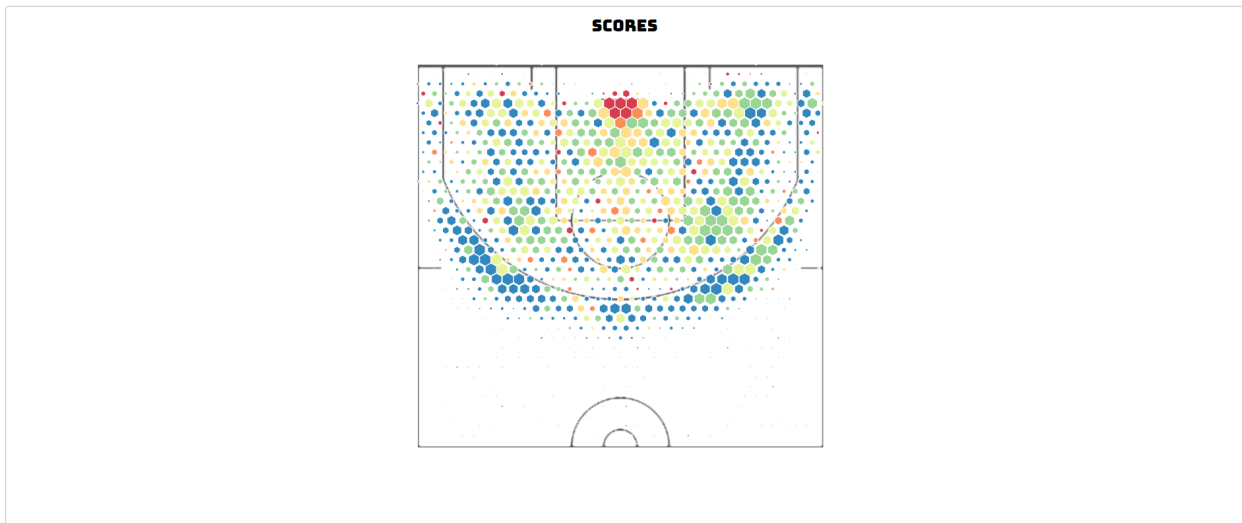


Figure 39 Change color map

CS6630 Final Project Process Book

After that, we added bar charts to calculate field goal made and field goal attempted analysis horizontally and vertically. They are top and right bars in figure 40.

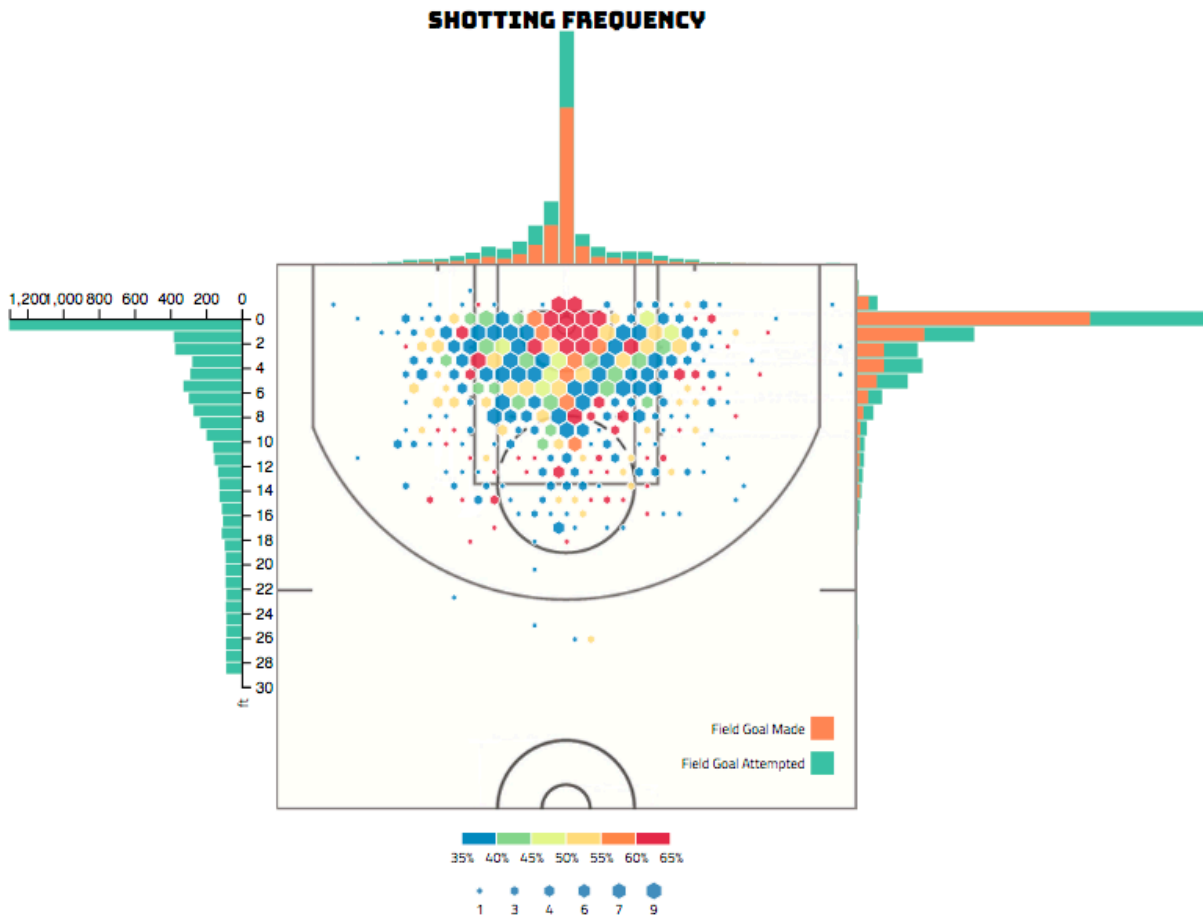


Figure 40 Design of shoot frequency view

Moreover, we also calculated the field goal made and field goal attempted by radius to the basket, which is displayed in left.

We users hover over the shoot frequency, a tooltip that shows the number of shoot and the field goal percentage as figure 41.

CS6630 Final Project Process Book

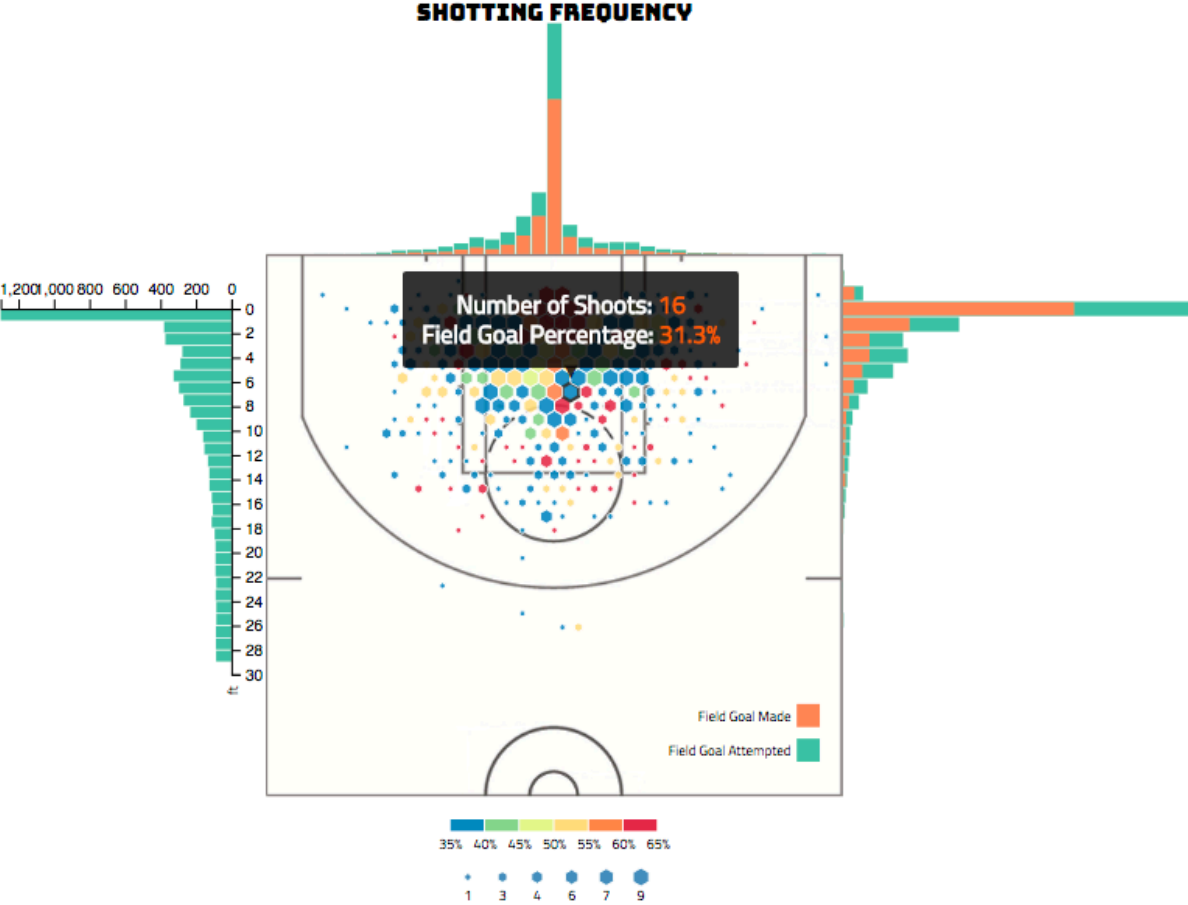


Figure 41 Interaction on shoot frequency view

And the number of field goal made and field goal attempted would display when users hover the top and right bars. Furthermore, if users hover the left bars, the radius position of calculating would be displayed on the chart as figure 42.

CS6630 Final Project Process Book

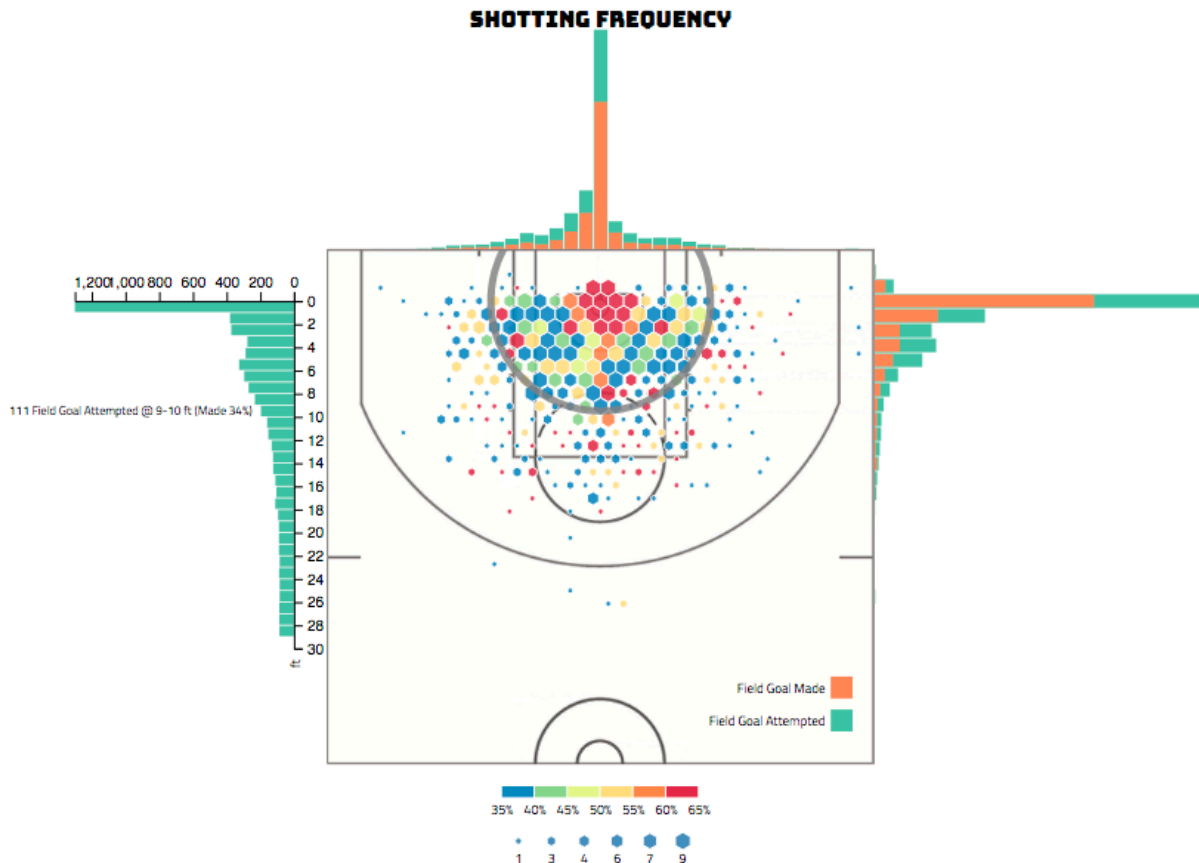


Figure 42 Interaction on shoot frequency view

10.6 Compare View

The first design of compare view as figure 43. After considering the fact that different attributes have different range of data values, we changed design views of comparison requirement. For example, points attribute could range between 20 and 30, however, the value of turnover attribute may be less than 1. And there are some attributes are percentage, such as field goal attempted percentage. It is unreasonable to encode the data with the same scale. Moreover, in the previous designs, it is hard for users to compare the length of horizontal bars. Therefore, we designed two more views as figure 44 shows that.

CS6630 Final Project Process Book

Compare two players.

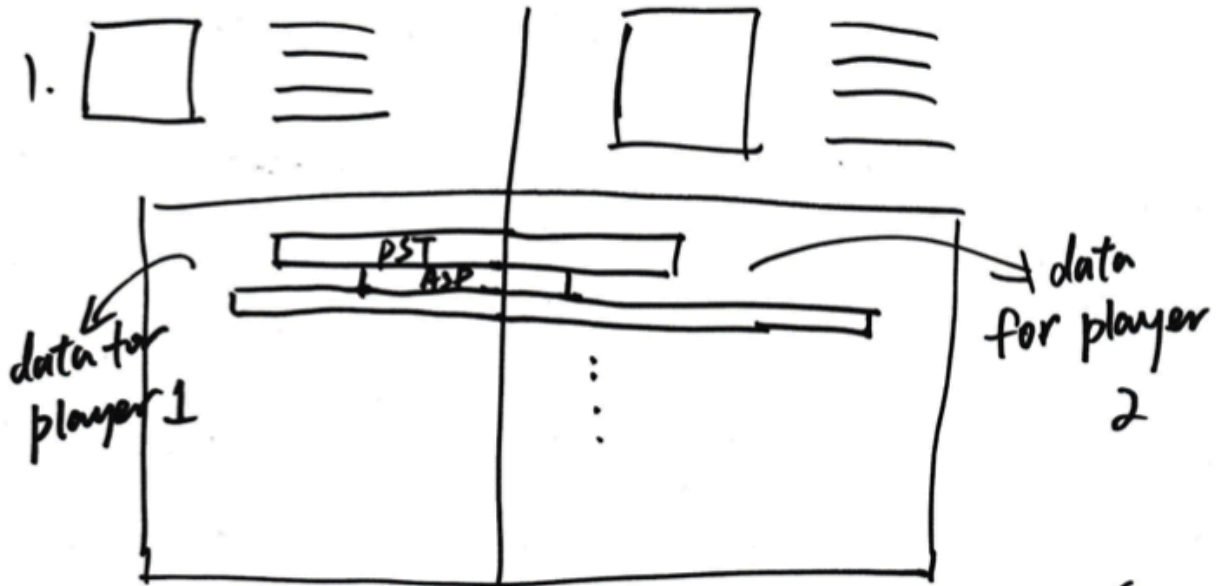


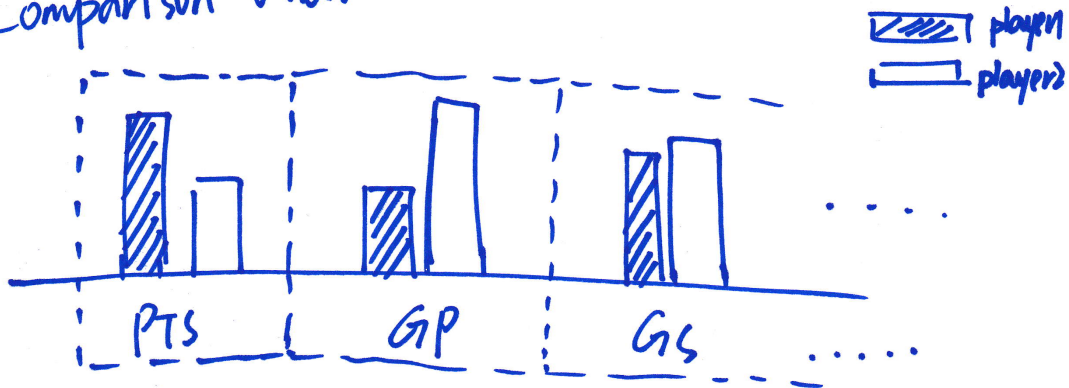
Figure 43 First design of compare view

In the first design showed in figure 44, we separated different attributes by grouping a dash box. And the advantage of this view is easy to compare two players' performance for each attribute. The second view design is set a fixed width rectangular. Then encode each attribute's data by the proportion of two players' performance data.

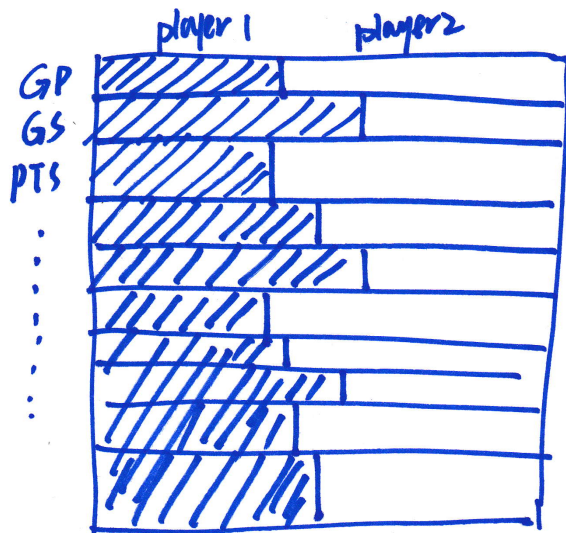
Therefore, we decided to use the second design. Because it fixed the problem of scaling we mentioned before and it is also easy to compare two players' performance.

CS6630 Final Project Process Book

Comparison View.



legend each scale., and value.



scale different attribute to the same notch of rest.

Figure 44 More design views for comparison

CS6630 Final Project Process Book

For interaction, we added tooltip for each bar to display the information of the player and the actual data value showed in figure 45.

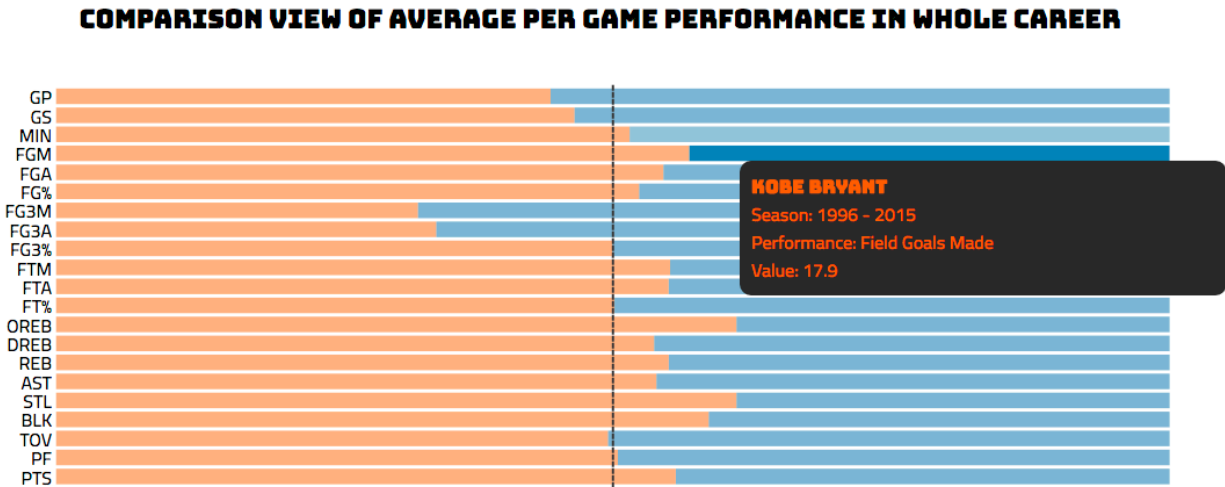


Figure 45 Interaction on Comparison view.

10. Implementation

10.1 Implementation on Milestone

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.

CS6630 Final Project Process Book

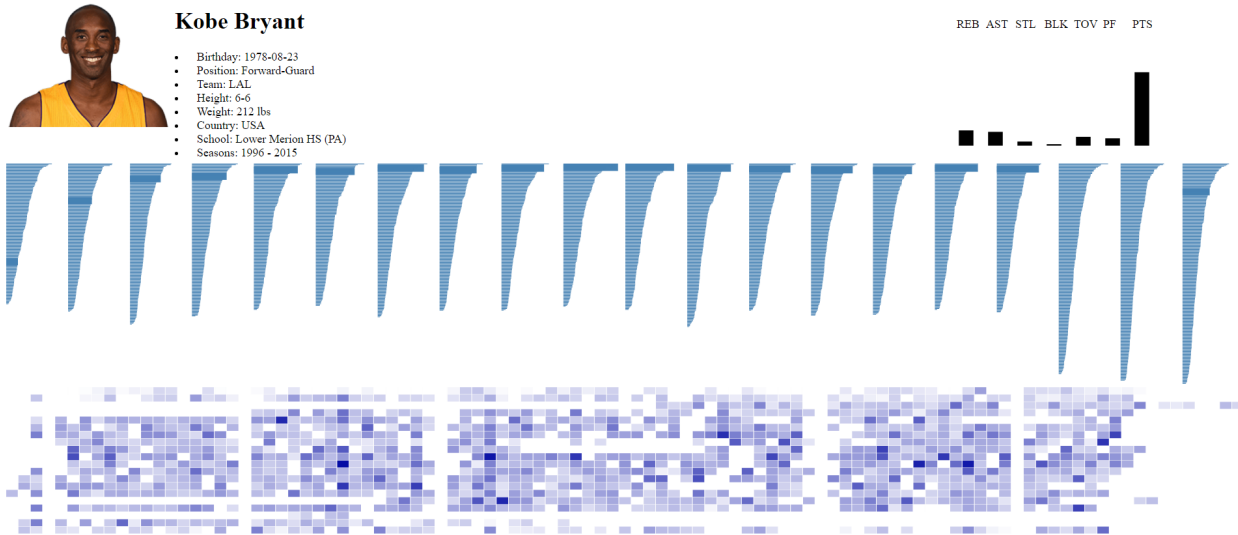



Figure 45 Website Prototype

10.2 Final implementation

Figure 46 shows our final implementation of main page and figure 47 is comparison page. Design process could be found in Design Evolution section. We separately implemented these two parts and combine them together. At last, we went through all style tooltip to make sure they are consistent.

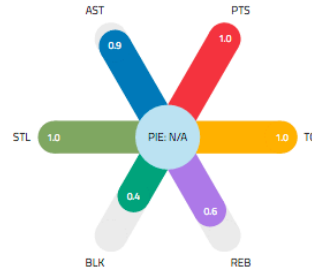
CS6630 Final Project Process Book

Change Player
Compare



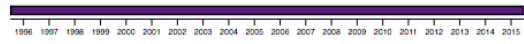
KOBE BRYANT

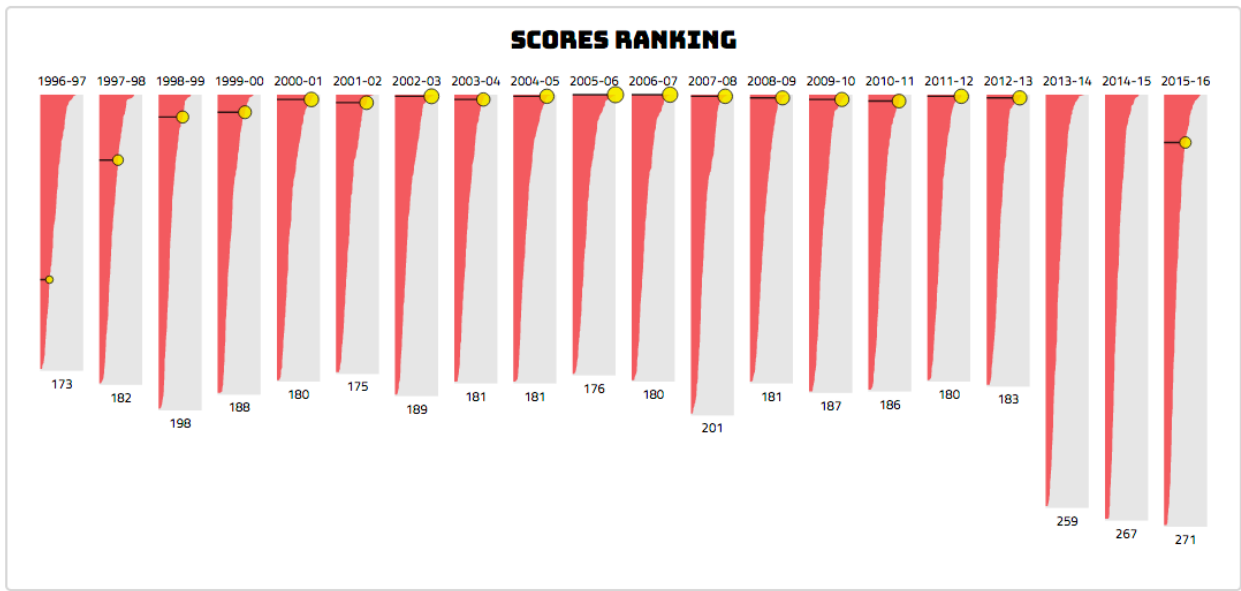
Team: Los Angeles Lakers
 Position: Forward-Guard
 Height: 6-6 ft
 Weight: 212 lbs
 Birthday: 1978-08-23
 Experience: 19 years
 Prior School: Lower Merion HS (PA)
 Seasons: 1996 - 2015
 Jersey: 24
 All Star Appearance: 15



OFFENSIVE

DEFENSIVE





CS6630 Final Project Process Book

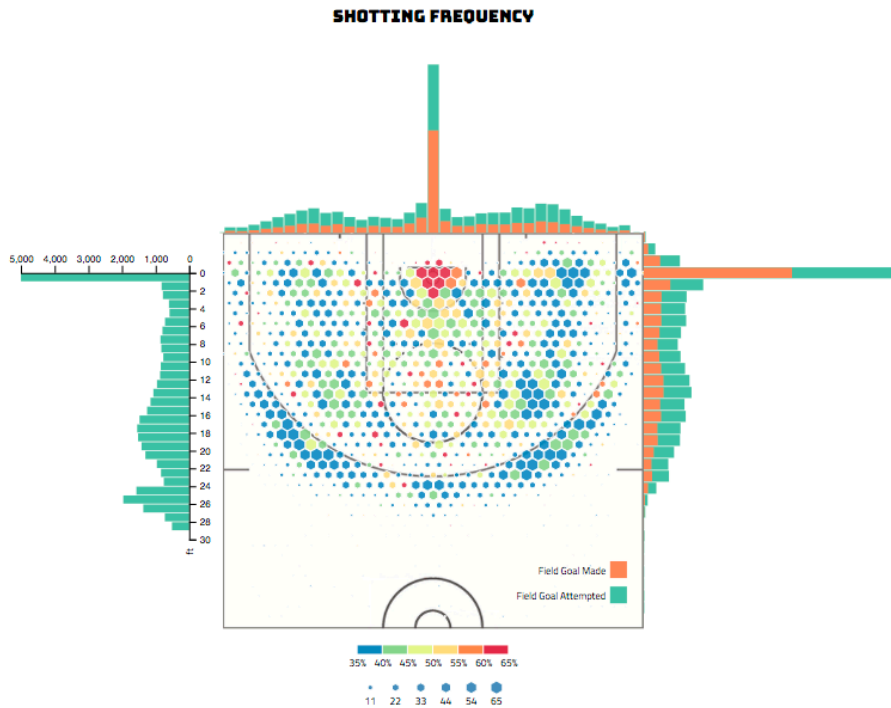
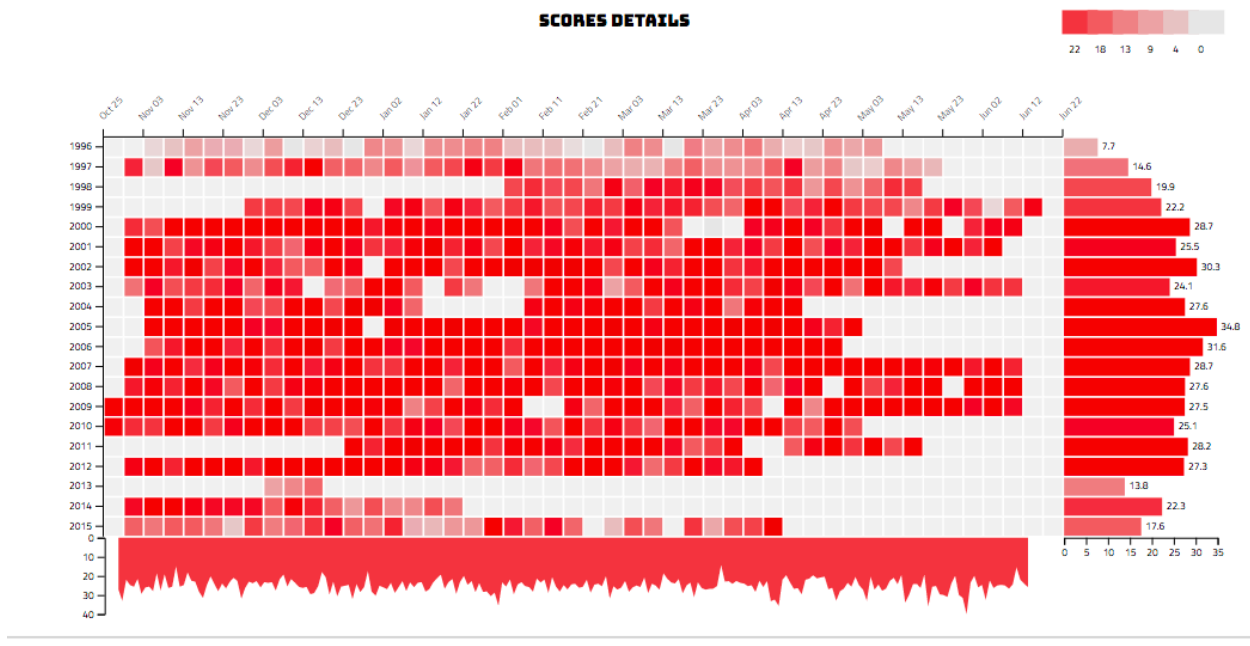



Figure 46 Final implementation of website page

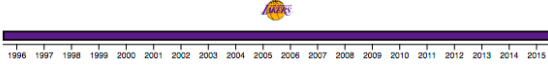
CS6630 Final Project Process Book


Change Player Single Player



KOBE BRYANT

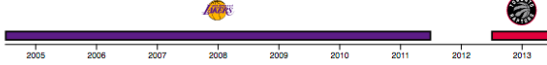
Team: Los Angeles Lakers
 Position: Forward-Guard
 Height: 6-6 ft
 Weight: 212 lbs
 Birthday: 1978-08-23
 Experience: 19 years
 Prior School: Lower Merion HS (PA)
 Seasons: 1996 - 2015
 Jersey: 24
 All Star Appearance: 15





ANDREW BYNUM

Team: Indiana Pacers
 Position: Center
 Height: 7-0 ft
 Weight: 285 lbs
 Birthday: 1987-10-27
 Experience: 7 years
 Prior School: St. Joseph HS (NJ)
 Seasons: 2005 - 2013
 Jersey: 17
 All Star Appearance: 1



COMPARISON VIEW OF AVERAGE PER GAME PERFORMANCE IN WHOLE CAREER

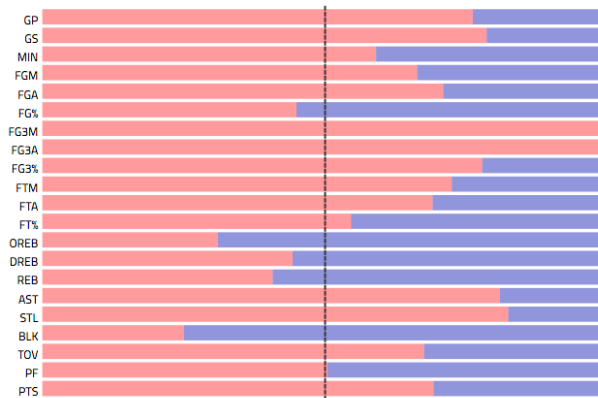


Figure 48 Final implementation of comparison view

CS6630 Final Project Process Book

10.2.1 Filter

The filter visualization part is response to the user's action when they click change player and compare button. The filter board will let users to choose the certain player according to team, position, time, all stars and name. When user's mouse sweeps over the name, it will highlight the player's name.

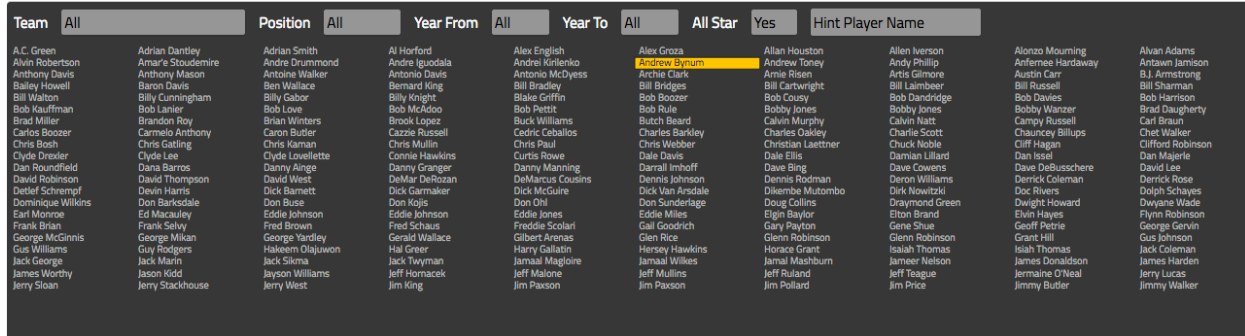


Figure 49 Implementation of filter

10.2.2 Basic information View

Figure 50 is the implementation of basic information view. In this visualization part, users could know basic personal information such as height, weight and birthday, etc. And from year time line, users could know the player's transfer information. On the left top of this board, two button "Change Player" and "Compare" will navigate our users to change another player or compare current player with another player. If user chooses compare button, the web will lead to the filter board and let user to choose the compared player's name. For example, if user want to compare Bill and Kobe, the chart will change to compare page of these two players.

On the right of the view is radial column chart shows this player's average performance on TOV, PTS, BLK, REB, STL and AST in his playing career. There is two tooltips over radial column chart. When users hover the text of attribute, the meaning of attribute and the scale information is displayed. If users hover each bar, an actual average value of the attribute would display. In addition, according to the player's average performance, whether the player is more offensive or more defensive is shown in the offense-defense balance bar. Each end side of the balance bar represent the full score of offense and defense. Implementation of these interactions are in figure 51 and figure 52.

CS6630 Final Project Process Book

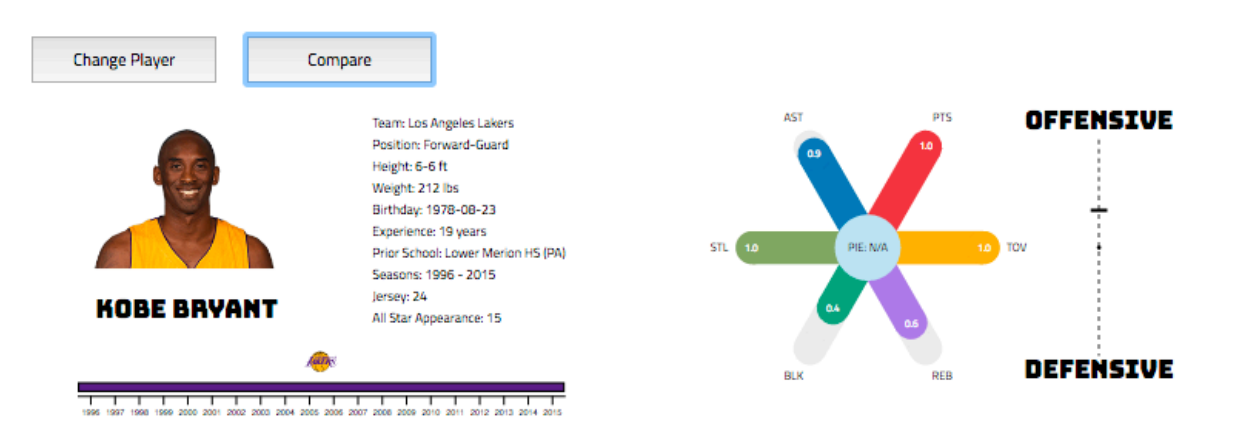


Figure 50 Implementation of basic information view

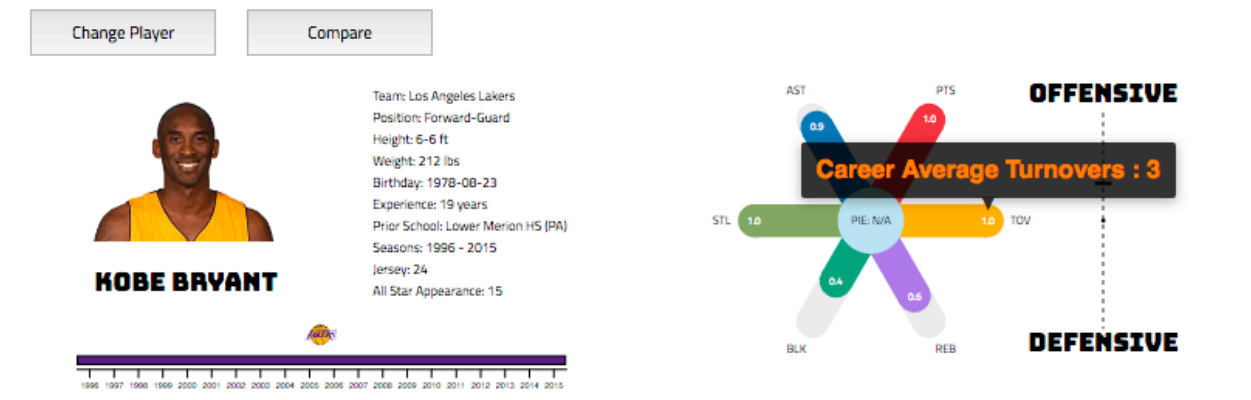


Figure 51 Implementation of interaction on radial column chart

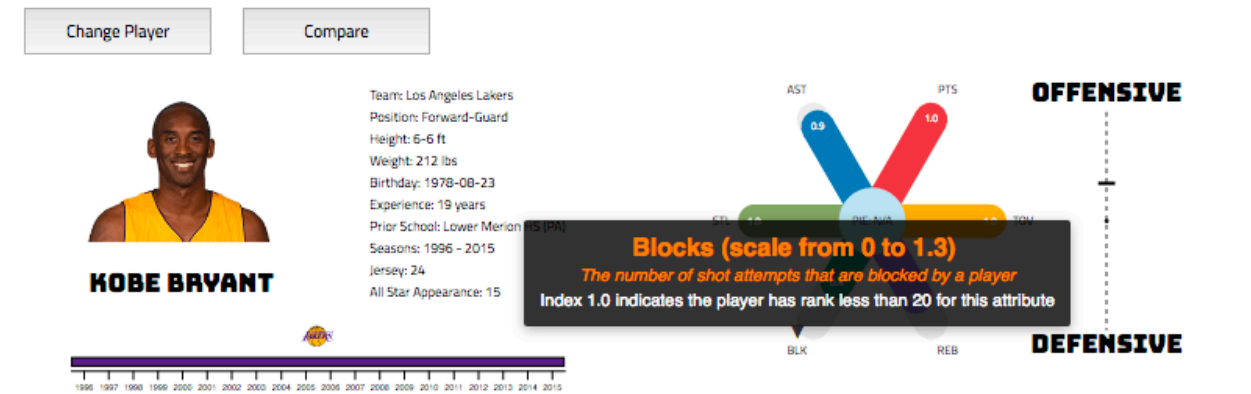


Figure 52 Implementation of interaction on radial column chart

CS6630 Final Project Process Book

Moreover, users could ask for ranking and detailed data for each attribute by clicking on the text in the radial column chart. For example, if users click on BLK, ranking chart and detailed data mesh could change to figure 53.

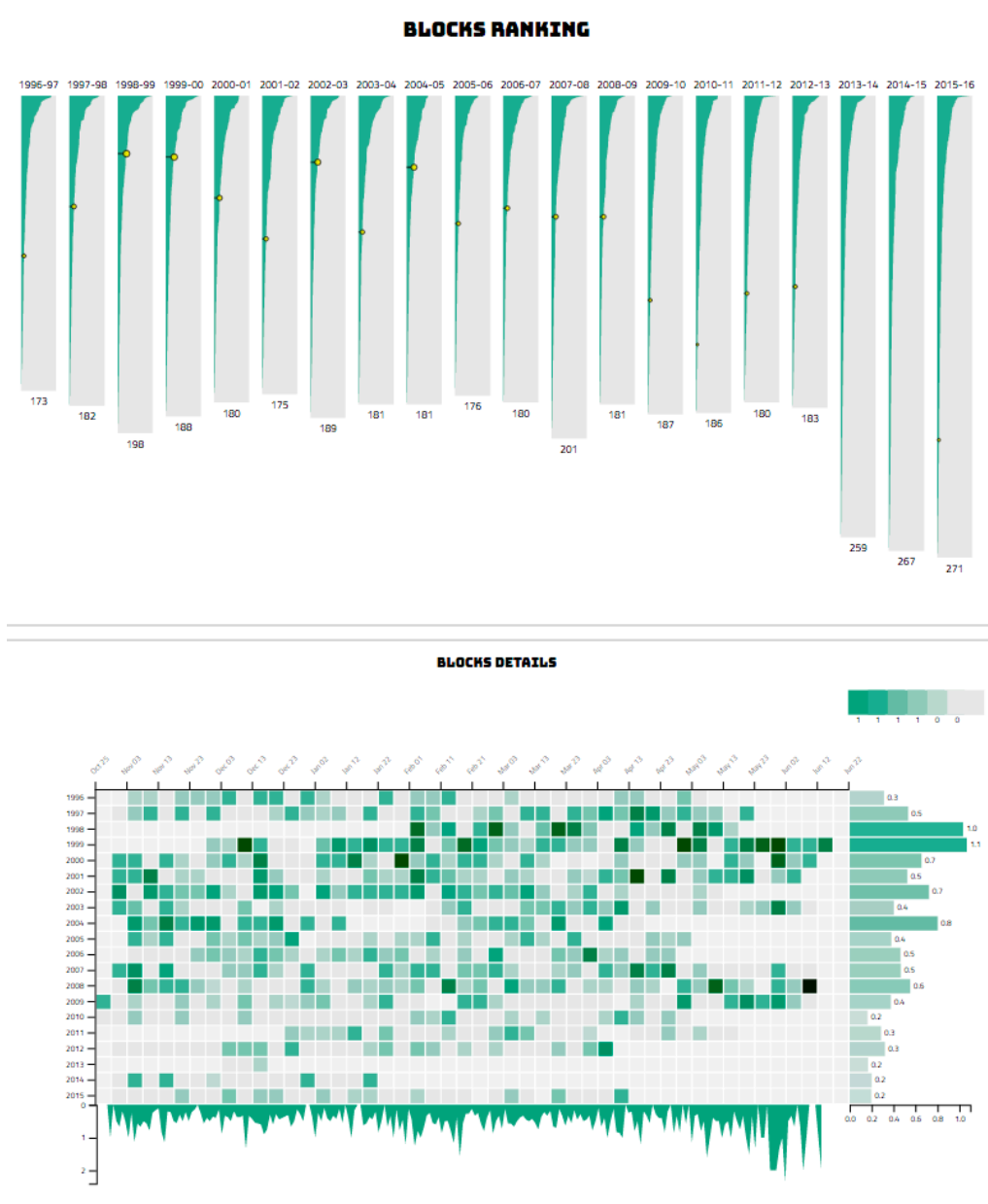


Figure 53 Implementation after users click on 'BLK' attribute

CS6630 Final Project Process Book

Also, there is a brush on the year time line. When users brush an interval of year, ranking chart and detailed data mesh would just keep data of brushed year as figure 54.

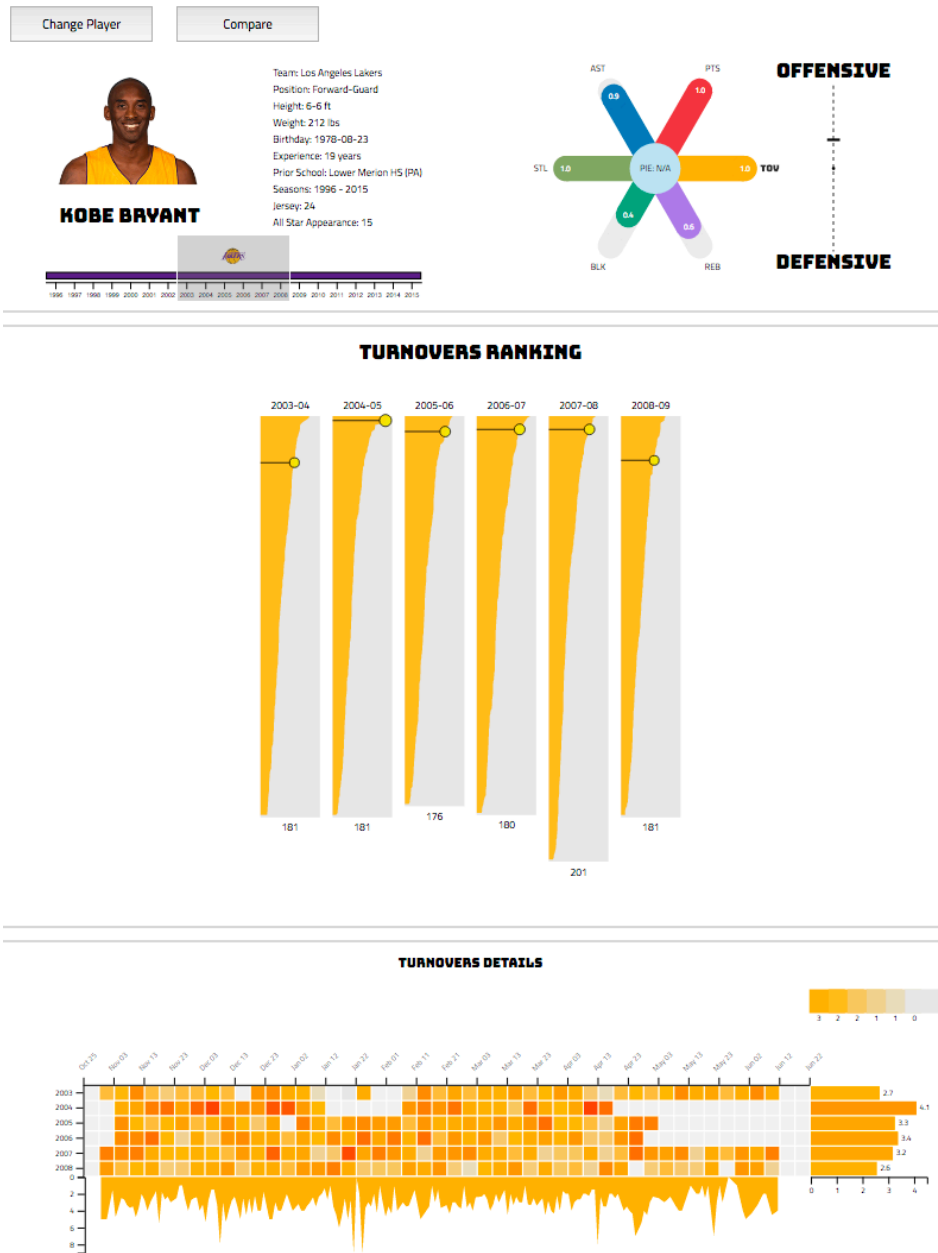


Figure 54 Implementation after brush year

CS6630 Final Project Process Book

10.2.3 Ranking View

In this visualization part, we show certain player's different performance attributes ranking in selected years. This graph is also responded to brush board. Besides, certain player's ranking information is highlighted by yellow circles. We also add the tooltip to display the player's score (the one with the black star) and his neighbor player's performance score. General implementation view and tooltip interaction shows in figure 55.

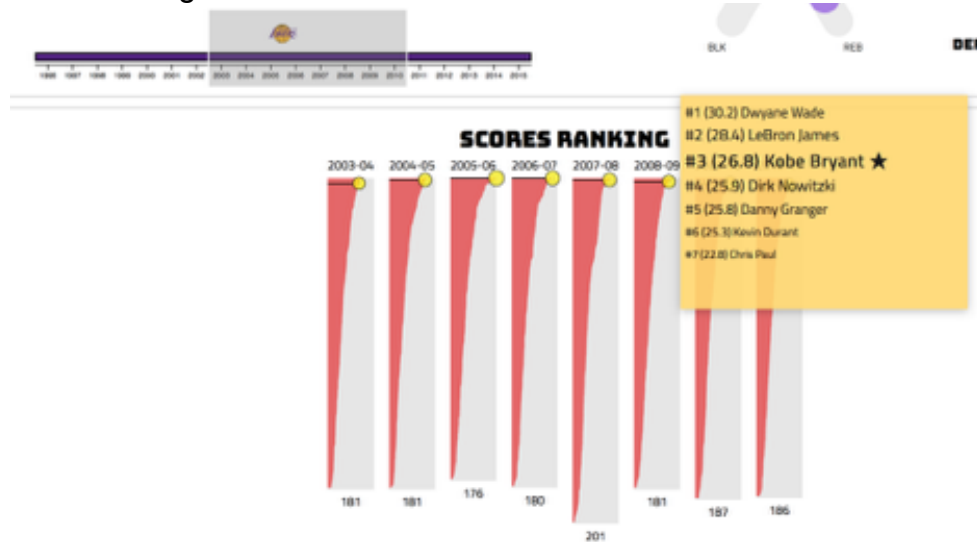


Figure 55 Implementation of ranking view and interaction

Also, if user move mouse over the ranking lines, they'll see some text in that tells them each player's performance score and his neighbor's ranking score. When they move their mouse away, the text fades away.

10.2.4 Detailed data view

In scores details graph, we mainly want to exhibit the player's performance as time goes by. In 2D histogram with proper bins, the temporal correlation of a player's performance on each attribute is displayed. This graph is also a response after user select certain time period on the brush board. Moreover, we want to tell our users this player's performance changing within certain years. That's why we surround the performance by adding two axis: X-Axis indicates the month and date when the game was played while the Y-Axis indicates the year value.

As the graph has showed, the histogram was visualized by a matrix, with average attribute value encoded by color. We also want to tell users the player's averaged performance change in terms of different periods of a season and different years. The bar chart on the right shows the histogram binned by year value, which elaborates the player's average performance for each season. The histogram on

CS6630 Final Project Process Book

the bottom shows the player's average performance during one single season as time goes by. In this way users can easily figure out a player's performance in different period of a season with this figure.

When user's mouse hover over each bins on the chart, it will show the game numbers represented by the small yellow circles with extra text explanation. The text board records the average scores that player's got within selected games. Also, the bar in the right side of the graph will response to the hover and highlight the total average score in this year. The bottom performance score is also responded with the hover and show the location and score value in selected month.

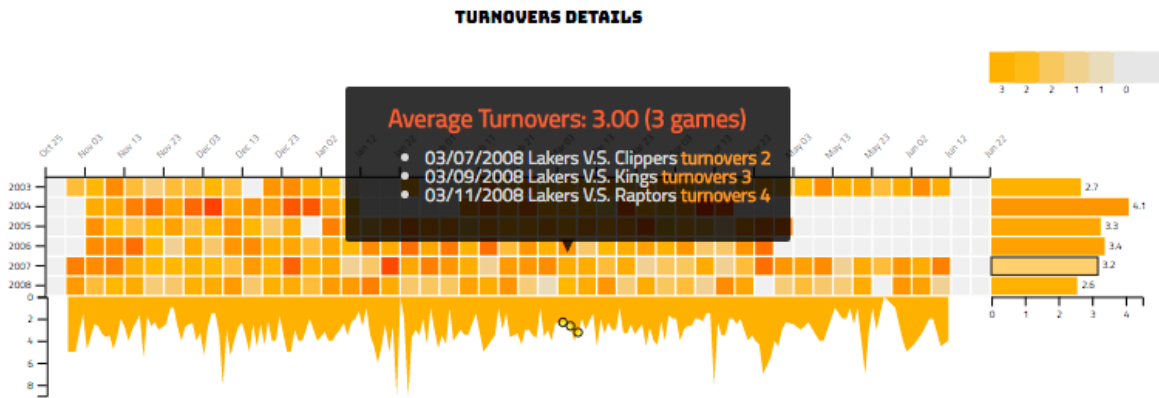


Figure 56 Implementation of detailed data mesh

10.2.5 Shoot frequency View

Figure 57 shows our implementation view of shoot frequency view. The function of shoot frequency view is to statistic field goal attempted(FGA), field goal made(FGM), and field goal percentage(FGP) with position of shooting. Each hexagon represents the total number of shoots and FPG. Colors are encoded by the value of FPG. The top and left bars represents FGM and FGA which statist horizontally and vertically over the field. Moreover, these data also statist by radius which is showed in the left of chart.

CS6630 Final Project Process Book

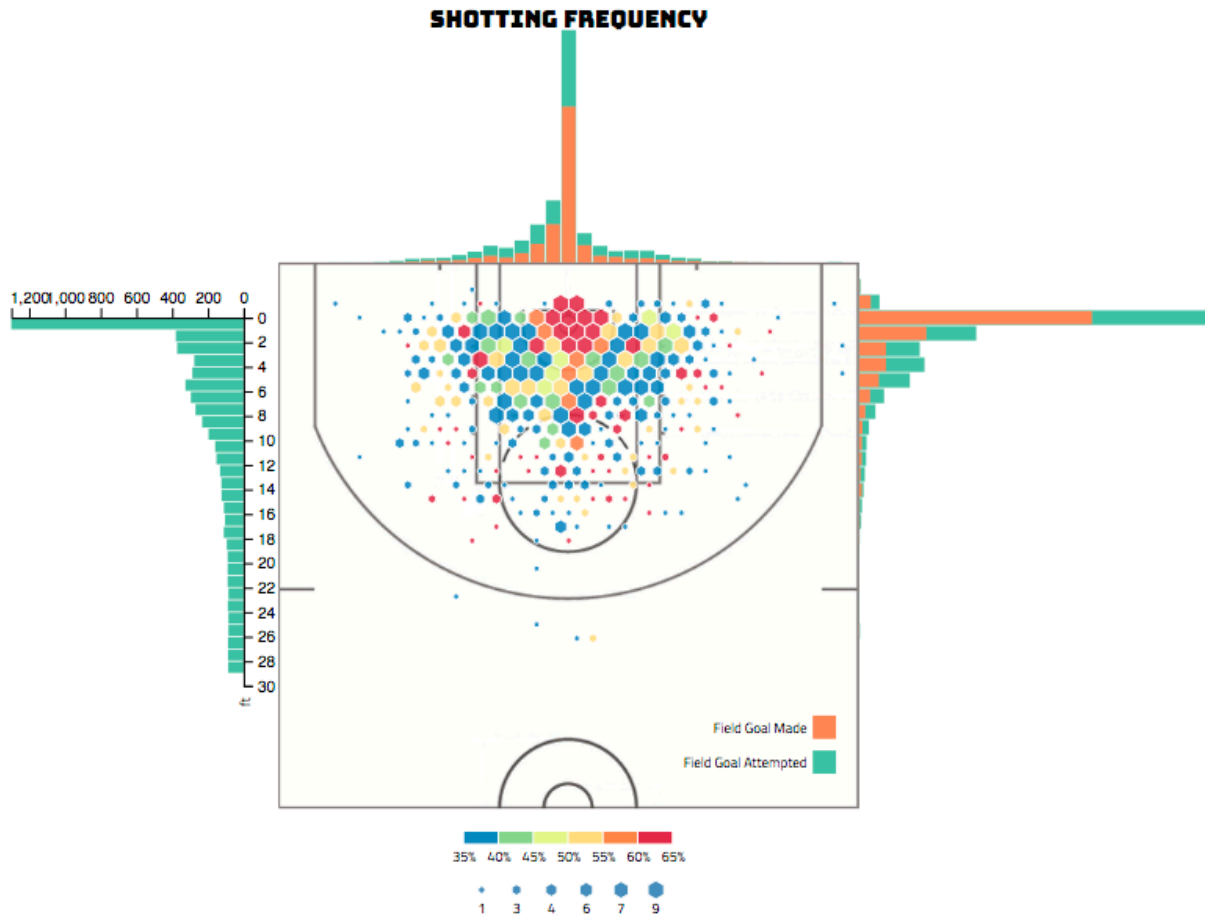


Figure 57 Implementation of shoot frequency chart

When users hover over the points on shooting frequency chart, a tooltip with information of the total number of shoot and FGP would be displayed. And when users hover the top and right bars, value of FGA and FGP would be displayed. When users hover the left bars, values and FGA and the distance to the basket would be displayed. All interactions are showed in figure 58, figure 59 and figure 60.

CS6630 Final Project Process Book

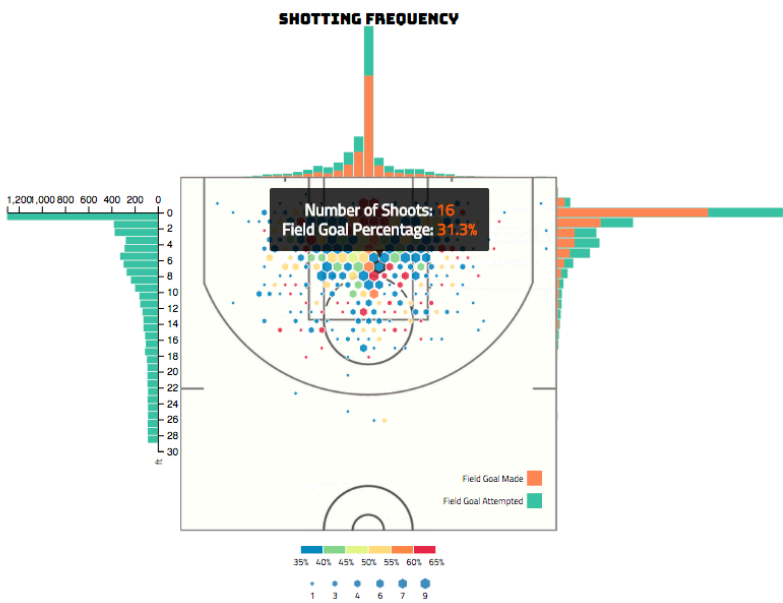


Figure 58 Interaction of shoot frequency chart

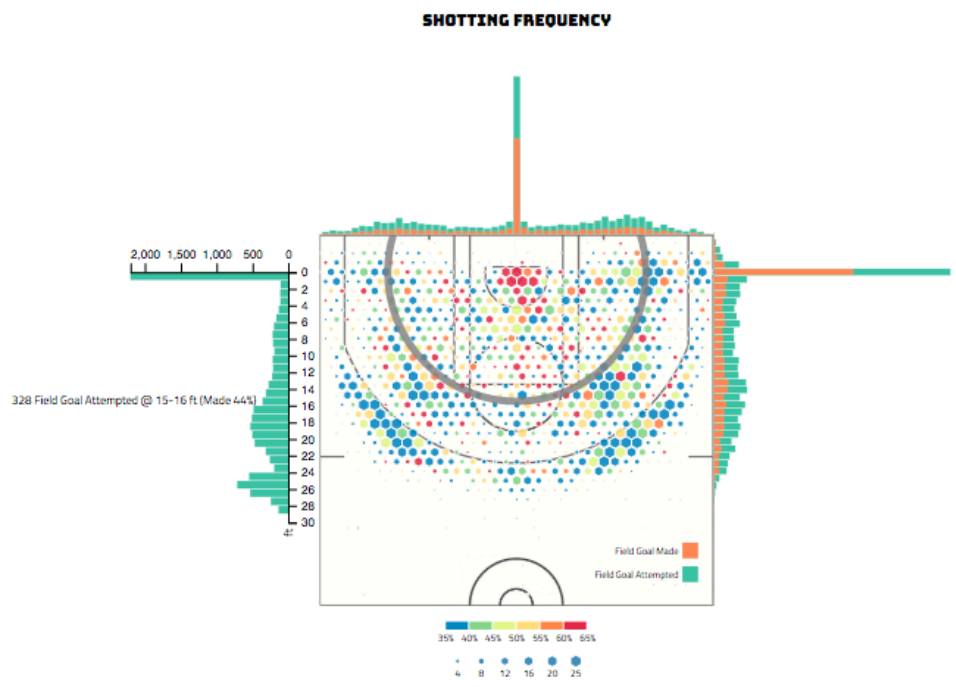


Figure 59 Interaction of shoot frequency chart

CS6630 Final Project Process Book

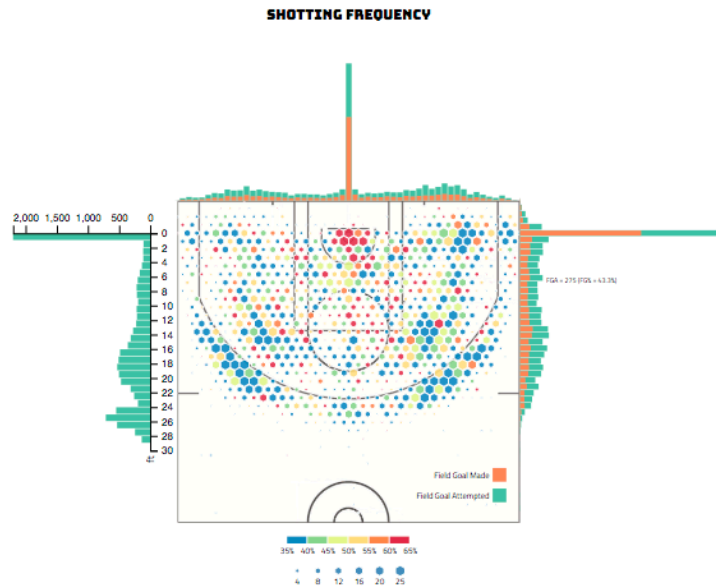



Figure 60 Interaction of shoot frequency chart

10.2.6 Compare View

Figure 61 shows our comparison view page. After users click on compare button and choose a compared player, this comparison view would be displayed. On the top of the view, two players' basic information is displayed. The comparison result is designed by our final view as we described above. Two different colors represent different players and each row represents an attribute. Users could compare two players' performance by comparing different skills, such as game played, game started, points, field goals or turnover, etc. Each skill's name is on the left side of chart.

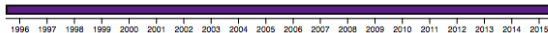
CS6630 Final Project Process Book


Change Player Single Player



KOBE BRYANT


Team: Los Angeles Lakers
 Position: Forward-Guard
 Height: 6-6 ft
 Weight: 212 lbs
 Birthday: 1978-08-23
 Experience: 19 years
 Prior School: Lower Merion HS (PA)
 Seasons: 1996 - 2015
 Jersey: 24
 All Star Appearance: 15





ANDREW BYNUM

Team: Indiana Pacers
 Position: Center
 Height: 7-0 ft
 Weight: 285 lbs
 Birthday: 1987-10-27
 Experience: 7 years
 Prior School: St. Joseph HS (NJ)
 Seasons: 2005 - 2013
 Jersey: 17
 All Star Appearance: 1



COMPARISON VIEW OF AVERAGE PER GAME PERFORMANCE IN WHOLE CAREER

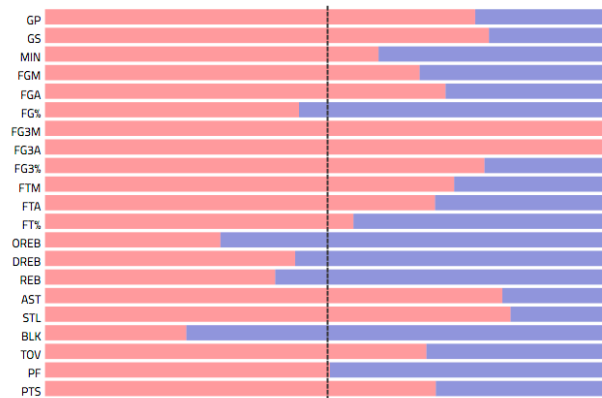


Figure 61 Implementation of comparison view

When users hover the mouse over the name of attributes, the full name of each attribute would display, as figure 62 shows that. Moreover, when users hover the mouse over each bar, a tooltip with information of player's name, career season, the name of attribute, and the specific value would appear, as figure 63 shows that.

CS6630 Final Project Process Book

COMPARISON VIEW OF AVERAGE PER GAME PERFORMANCE IN WHOLE CAREER



Figure 62 Interaction of tooltip shows full name of header

COMPARISON VIEW OF AVERAGE PER GAME PERFORMANCE IN WHOLE CAREER

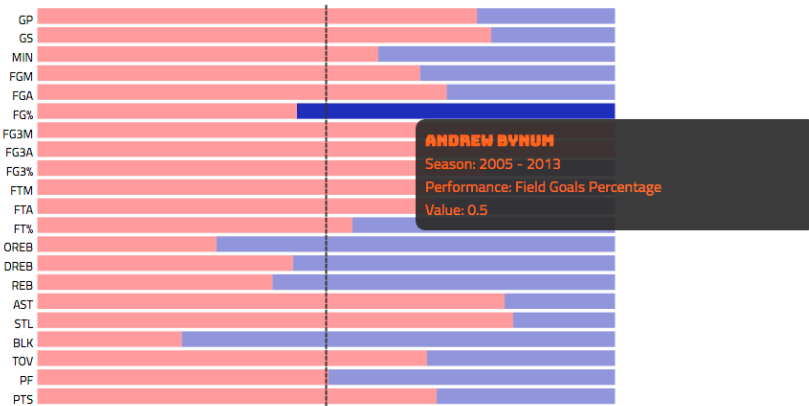


Figure 63 Interaction of tooltip over bars

CS6630 Final Project Process Book

11. Evaluation and Conclusion

From our visualization, users could obtain players' performance statistics and compare two different players' performance in their career. All questions from section 4 could be answered. For example, users could know a player's ranking of six important attributes, such as points, field goal or turnover.

We think our visualization works well. We achieved our tasks and must have features. We also added clear legend and tooltip to improve understanding. Furthermore, we consisted the style of tooltip so that all views in our visualization would be consistent. We make sure users could obtain enough and important information they need.

However, in our designs, there are several details need to be fixed in the future. First, in shoot chart, there are too much information, we may need to add some description to let users understand it quicker. Second, we have a lot of data in our project, some players' data is not enough to display views. We need to design a more effective filter to search players. Furthermore, although we considered the different monitors' resolution and zoom up/in pages, and our website works for these conditions mostly. However, there is also some zoom issues because our ineffective dealing method. In order to improve our design, we also desire to finish our optional feature to do customized team-up.

Reference:

[1] <https://github.com/d3/d3-plugins/tree/master/hexbin>

[2] <http://colorbrewer2.org/#type=sequential&scheme=BuGn&n=3>