EURO 2024 TÜRKİYE

In [987…
```
#![Resim asdasdasdasd](trmil.jpg)
```

In [3]:
```
import warnings
warnings.filterwarnings("ignore")
```

In [5]:
```
from statsbombpy import sb
import pandas as pd
from mplsoccer import VerticalPitch,Pitch
from highlight_text import ax_text, fig_text
from matplotlib.colors import LinearSegmentedColormap
import matplotlib.pyplot as plt
import matplotlib.patheffects as path_effects
import seaborn as sns
```

In [13]:
```
import highlight_text
print(dir(highlight_text))
```

```
['HighlightText', '__builtins__', '__cached__', '__doc__', '__file__', '__loader__', '__name__', '__package__',
'__path__', '__spec__', 'ax_text', 'fig_text', 'htext']
```

In [7]:
```
free_comps = sb.competitions()
free_comps.head()
```

Out[7]:

	competition_id	season_id	country_name	competition_name	competition_gender	competition_youth	competition_international	season_nar
0	9	281	Germany	1. Bundesliga	male	False	False	2023/20
1	9	27	Germany	1. Bundesliga	male	False	False	2015/20
2	1267	107	Africa	African Cup of Nations	male	False	True	20
3	16	4	Europe	Champions League	male	False	False	2018/20
4	16	1	Europe	Champions League	male	False	False	2017/20

In [9]:
```
free_comps["competition_name"].value_counts()
```

Out[9]:
```
competition_name
Champions League           18
La Liga                    18
FIFA World Cup              8
Ligue 1                     3
Copa del Rey                3
FA Women's Super League     3
1. Bundesliga               2
UEFA Euro                   2
Serie A                     2
Premier League              2
Liga Profesional            2
Women's World Cup           2
African Cup of Nations      1
Major League Soccer         1
North American League       1
NWSL                        1
Indian Super league         1
FIFA U20 World Cup          1
Copa America                1
UEFA Europa League          1
UEFA Women's Euro           1
Name: count, dtype: int64
```

In [11]:
```
euro_2024_matches = sb.matches(competition_id=55, season_id=282)
euro_2024_matches.head(20)
```

Out[11]:

	match_id	match_date	kick_off	competition	season	home_team	away_team	home_score	away_score	match_status	...	last_upd
0	3942819	2024-07-10	22:00:00.000	Europe - UEFA Euro	2024	Netherlands	England	1	2	available	...	15T07:57:
1	3943043	2024-07-14	22:00:00.000	Europe - UEFA Euro	2024	Spain	England	2	1	available	...	15T15:52:
2	3942752	2024-07-09	22:00:00.000	Europe - UEFA Euro	2024	Spain	France	2	1	available	...	10T13:01:
3	3942382	2024-07-06	22:00:00.000	Europe - UEFA Euro	2024	Netherlands	Turkey	2	1	available	...	10T06:56:
4	3942349	2024-07-05	22:00:00.000	Europe - UEFA Euro	2024	Portugal	France	0	0	available	...	12T02:00:
5	3930180	2024-06-25	19:00:00.000	Europe - UEFA Euro	2024	Netherlands	Austria	2	3	available	...	11T13:25:
6	3930171	2024-06-20	19:00:00.000	Europe - UEFA Euro	2024	Denmark	England	1	1	available	...	12T05:19:
7	3942227	2024-07-06	19:00:00.000	Europe - UEFA Euro	2024	England	Switzerland	1	1	available	...	14T15:57:
8	3942226	2024-07-05	19:00:00.000	Europe - UEFA Euro	2024	Spain	Germany	2	1	available	...	13T03:52:
9	3938645	2024-06-26	19:00:00.000	Europe - UEFA Euro	2024	Ukraine	Belgium	0	0	available	...	11T16:17:
10	3930184	2024-06-26	22:00:00.000	Europe - UEFA Euro	2024	Czech Republic	Turkey	1	2	available	...	10T11:03:
11	3941022	2024-07-02	22:00:00.000	Europe - UEFA Euro	2024	Austria	Turkey	1	2	available	...	10T11:31:
12	3941021	2024-07-02	19:00:00.000	Europe - UEFA Euro	2024	Romania	Netherlands	0	3	available	...	12T23:28:
13	3941020	2024-07-01	22:00:00.000	Europe - UEFA Euro	2024	Portugal	Slovenia	0	0	available	...	13T19:48:
14	3941019	2024-07-01	19:00:00.000	Europe - UEFA Euro	2024	France	Belgium	1	0	available	...	10T11:04:
15	3941018	2024-06-30	22:00:00.000	Europe - UEFA Euro	2024	Spain	Georgia	4	1	available	...	12T06:08:
16	3941017	2024-06-30	19:00:00.000	Europe - UEFA Euro	2024	England	Slovakia	2	1	available	...	10T11:31:
17	3930182	2024-06-25	22:00:00.000	Europe - UEFA Euro	2024	Denmark	Serbia	0	0	available	...	10T09:19:
18	3930179	2024-06-24	22:00:00.000	Europe - UEFA Euro	2024	Albania	Spain	0	1	available	...	01T06:07:
19	3940983	2024-06-29	22:00:00.000	Europe - UEFA Euro	2024	Germany	Denmark	2	0	available	...	07T11:16:

20 rows × 22 columns

In [13]:
```
team = "Turkey"
matches_df = euro_2024_matches[(euro_2024_matches["home_team"] == team) | (euro_2024_matches["away_team"] == te
matches_df = matches_df.sort_values(by="match_date",ascending=False)
```

In [17]:
```
matches_df
```

Out[17]:

	match_id	match_date	kick_off	competition	season	home_team	away_team	home_score	away_score	match_status	...	last_upd
3	3942382	2024-07-06	22:00:00.000	Europe - UEFA Euro	2024	Netherlands	Turkey	2	1	available	...	10T06:56:3
11	3941022	2024-07-02	22:00:00.000	Europe - UEFA Euro	2024	Austria	Turkey	1	2	available	...	10T11:31:1
10	3930184	2024-06-26	22:00:00.000	Europe - UEFA Euro	2024	Czech Republic	Turkey	1	2	available	...	10T11:03:5
42	3930174	2024-06-22	19:00:00.000	Europe - UEFA Euro	2024	Turkey	Portugal	0	3	available	...	11T16:43:4
32	3938639	2024-06-18	19:00:00.000	Europe - UEFA Euro	2024	Turkey	Georgia	3	1	available	...	12T06:05:1

5 rows × 22 columns

In [15]:
```
# latest_match_id = matches_df.match_id.iloc[0]    son maç için
```

In [17]:
```
#events_df = sb.events(match_id = latest_match_id )
#events_df.head(5)
```

Out[17]:

	50_50	bad_behaviour_card	ball_receipt_outcome	ball_recovery_recovery_failure	block_deflection	block_save_block	carry_end_location	cl
0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
1	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
2	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
3	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
4	NaN	NaN	NaN	NaN	NaN	NaN	NaN	

5 rows × 93 columns

In [19]:
```
import pandas as pd

# Türkiye'nin maç ID'leri
match_ids = [3942382, 3941022, 3930184, 3930174, 3938639]

all_events = []

for match_id in match_ids:
    events_df = sb.events(match_id=match_id)
    all_events.append(events_df)

events_all_df = pd.concat(all_events, ignore_index=True)

print(events_all_df.head(5))
```

```
  50_50 bad_behaviour_card ball_receipt_outcome  \
0   NaN               NaN                  NaN
1   NaN               NaN                  NaN
2   NaN               NaN                  NaN
3   NaN               NaN                  NaN
4   NaN               NaN                  NaN

  ball_recovery_recovery_failure block_deflection block_save_block  \
0                            NaN              NaN              NaN
1                            NaN              NaN              NaN
2                            NaN              NaN              NaN
3                            NaN              NaN              NaN
4                            NaN              NaN              NaN

  carry_end_location clearance_aerial_won clearance_body_part clearance_head  \
0                NaN                  NaN                 NaN            NaN
1                NaN                  NaN                 NaN            NaN
2                NaN                  NaN                 NaN            NaN
3                NaN                  NaN                 NaN            NaN
4                NaN                  NaN                 NaN            NaN

   ... dribble_overrun pass_outswinging shot_one_on_one clearance_other  \
0  ...             NaN              NaN             NaN             NaN
1  ...             NaN              NaN             NaN             NaN
2  ...             NaN              NaN             NaN             NaN
3  ...             NaN              NaN             NaN             NaN
4  ...             NaN              NaN             NaN             NaN

  pass_cut_back pass_no_touch shot_deflected  ball_recovery_offensive  \
0           NaN           NaN            NaN                      NaN
1           NaN           NaN            NaN                      NaN
2           NaN           NaN            NaN                      NaN
3           NaN           NaN            NaN                      NaN
4           NaN           NaN            NaN                      NaN

  pass_miscommunication dribble_no_touch
0                   NaN              NaN
1                   NaN              NaN
2                   NaN              NaN
3                   NaN              NaN
4                   NaN              NaN

[5 rows x 104 columns]
```

In [21]:
```
events_all_df.columns
```

Out[21]:
```
Index(['50_50', 'bad_behaviour_card', 'ball_receipt_outcome',
       'ball_recovery_recovery_failure', 'block_deflection',
       'block_save_block', 'carry_end_location', 'clearance_aerial_won',
       'clearance_body_part', 'clearance_head',
       ...
       'dribble_overrun', 'pass_outswinging', 'shot_one_on_one',
       'clearance_other', 'pass_cut_back', 'pass_no_touch', 'shot_deflected',
       'ball_recovery_offensive', 'pass_miscommunication', 'dribble_no_touch'],
      dtype='object', length=104)
```

In [23]:
```
events_all_df.shape
```

Out[23]:
```
(17713, 104)
```

In [25]:
```
events_all_df[["x","y"]] = events_all_df["location"].apply(pd.Series)
events_all_df[["pass_end_x","pass_end_y"]] = events_all_df["pass_end_location"].apply(pd.Series)
events_all_df[["carry_end_x","carry_end_y"]] = events_all_df["carry_end_location"].apply(pd.Series)
```

In [27]:
```
events_all_df.type.unique()
```

Out[27]:
```
array(['Starting XI', 'Half Start', 'Pass', 'Ball Receipt*', 'Carry',
       'Duel', 'Clearance', 'Pressure', 'Block', 'Ball Recovery',
       'Dribbled Past', 'Dribble', 'Shot', 'Goal Keeper', 'Dispossessed',
       'Interception', 'Miscontrol', 'Foul Committed', 'Foul Won',
       '50/50', 'Injury Stoppage', 'Player Off', 'Player On', 'Shield',
       'Half End', 'Substitution', 'Tactical Shift', 'Referee Ball-Drop',
       'Own Goal Against', 'Own Goal For', 'Bad Behaviour', 'Error',
       'Offside'], dtype=object)
```

In [29]:
```
f3rd_passes = events_all_df[(events_all_df.team == team) & (events_all_df.type == "Pass") & (events_all_df.x <

f3rd_passes_count = f3rd_passes.groupby("player").size().reset_index()
```

In [31]:
```
f3rd_passes_count
```

Out[31]:

	player	0
0	Abdülkerim Bardakcı	20
1	Arda Güler	11
2	Barış Alper Yılmaz	4
3	Fehmi Mert Günok	4
4	Ferdi Erenay Kadıoğlu	16
5	Hakan Çalhanoğlu	19
6	Kaan Ayhan	13
7	Kenan Yildiz	4
8	Mehmet Zeki Çelik	3
9	Merih Demiral	3
10	Mert Müldür	3
11	Muhammed Kerem Aktürkoğlu	4
12	Okay Yokuşlu	2
13	Orkun Kökçü	6
14	Salih Özcan	3
15	Samet Akaydin	19
16	Yusuf Yazıcı	1
17	İsmail Yüksek	8

In [33]:
```
f3rd_passes_count.rename(columns={f3rd_passes_count.columns[1]:"Passes"},inplace = True)
```

In [35]:
```
f3rd_passes_count
```

Out[35]:

	player	Passes
0	Abdülkerim Bardakcı	20
1	Arda Güler	11
2	Barış Alper Yılmaz	4
3	Fehmi Mert Günok	4
4	Ferdi Erenay Kadıoğlu	16
5	Hakan Çalhanoğlu	19
6	Kaan Ayhan	13
7	Kenan Yildiz	4
8	Mehmet Zeki Çelik	3
9	Merih Demiral	3
10	Mert Müldür	3
11	Muhammed Kerem Aktürkoğlu	4
12	Okay Yokuşlu	2
13	Orkun Kökçü	6
14	Salih Özcan	3
15	Samet Akaydin	19
16	Yusuf Yazıcı	1
17	İsmail Yüksek	8

In [37]:
```
f3rd_carries = events_all_df[(events_all_df.type == "Carry") & (events_all_df.x < 80 ) & (events_all_df.carry_e

f3rd_carries_count = f3rd_carries.groupby("player").size().reset_index()

f3rd_carries_count.rename(columns={f3rd_carries_count.columns[1]:"Carries"},inplace = True)
```

In [39]:
```
f3rd_carries_count
```

Out[39]:

	player	Carries
0	Abdülkerim Bardakcı	3
1	Arda Güler	6
2	Barış Alper Yılmaz	11
3	Cenk Tosun	1
4	Ferdi Erenay Kadıoğlu	16
5	Hakan Çalhanoğlu	5
6	Kaan Ayhan	2
7	Kenan Yildiz	7
8	Mehmet Zeki Çelik	1
9	Mert Müldür	11
10	Muhammed Kerem Aktürkoğlu	2
11	Okay Yokuşlu	1
12	Orkun Kökçü	3
13	Samet Akaydin	2
14	Yunus Akgün	3
15	İsmail Yüksek	2

In [41]:
```
progression_df = pd.merge(f3rd_passes_count, f3rd_carries_count, how="outer",on=["player"])
```

In [43]:
```
progression_df
```

Out[43]:

	player	Passes	Carries
0	Abdülkerim Bardakcı	20.0	3.0
1	Arda Güler	11.0	6.0
2	Barış Alper Yılmaz	4.0	11.0
3	Fehmi Mert Günok	4.0	NaN
4	Ferdi Erenay Kadıoğlu	16.0	16.0
5	Hakan Çalhanoğlu	19.0	5.0
6	Kaan Ayhan	13.0	2.0
7	Kenan Yildiz	4.0	7.0
8	Mehmet Zeki Çelik	3.0	1.0
9	Merih Demiral	3.0	NaN
10	Mert Müldür	3.0	11.0
11	Muhammed Kerem Aktürkoğlu	4.0	2.0
12	Okay Yokuşlu	2.0	1.0
13	Orkun Kökçü	6.0	3.0
14	Salih Özcan	3.0	NaN
15	Samet Akaydin	19.0	2.0
16	Yusuf Yazıcı	1.0	NaN
17	İsmail Yüksek	8.0	2.0
18	Cenk Tosun	NaN	1.0
19	Yunus Akgün	NaN	3.0

In [45]:
```
progression_df = progression_df.fillna(0)
```

In [47]:
```
progression_df
```

Out[47]:

	player	Passes	Carries
0	Abdülkerim Bardakcı	20.0	3.0
1	Arda Güler	11.0	6.0
2	Barış Alper Yılmaz	4.0	11.0
3	Fehmi Mert Günok	4.0	0.0
4	Ferdi Erenay Kadıoğlu	16.0	16.0
5	Hakan Çalhanoğlu	19.0	5.0
6	Kaan Ayhan	13.0	2.0
7	Kenan Yildiz	4.0	7.0
8	Mehmet Zeki Çelik	3.0	1.0
9	Merih Demiral	3.0	0.0
10	Mert Müldür	3.0	11.0
11	Muhammed Kerem Aktürkoğlu	4.0	2.0
12	Okay Yokuşlu	2.0	1.0
13	Orkun Kökçü	6.0	3.0
14	Salih Özcan	3.0	0.0
15	Samet Akaydin	19.0	2.0
16	Yusuf Yazıcı	1.0	0.0
17	İsmail Yüksek	8.0	2.0
18	Cenk Tosun	0.0	1.0
19	Yunus Akgün	0.0	3.0

In [49]:
```
progression_df["total"] = progression_df["Passes"] + progression_df["Carries"]
```

In [51]:
```
progression_df
```

Out[51]:

	player	Passes	Carries	total
0	Abdülkerim Bardakcı	20.0	3.0	23.0
1	Arda Güler	11.0	6.0	17.0
2	Barış Alper Yılmaz	4.0	11.0	15.0
3	Fehmi Mert Günok	4.0	0.0	4.0
4	Ferdi Erenay Kadıoğlu	16.0	16.0	32.0
5	Hakan Çalhanoğlu	19.0	5.0	24.0
6	Kaan Ayhan	13.0	2.0	15.0
7	Kenan Yildiz	4.0	7.0	11.0
8	Mehmet Zeki Çelik	3.0	1.0	4.0
9	Merih Demiral	3.0	0.0	3.0
10	Mert Müldür	3.0	11.0	14.0
11	Muhammed Kerem Aktürkoğlu	4.0	2.0	6.0
12	Okay Yokuşlu	2.0	1.0	3.0
13	Orkun Kökçü	6.0	3.0	9.0
14	Salih Özcan	3.0	0.0	3.0
15	Samet Akaydin	19.0	2.0	21.0
16	Yusuf Yazıcı	1.0	0.0	1.0
17	İsmail Yüksek	8.0	2.0	10.0
18	Cenk Tosun	0.0	1.0	1.0
19	Yunus Akgün	0.0	3.0	3.0

In [53]:
```
progression_df.sort_values(by = "total",ascending = False,inplace=True)
```

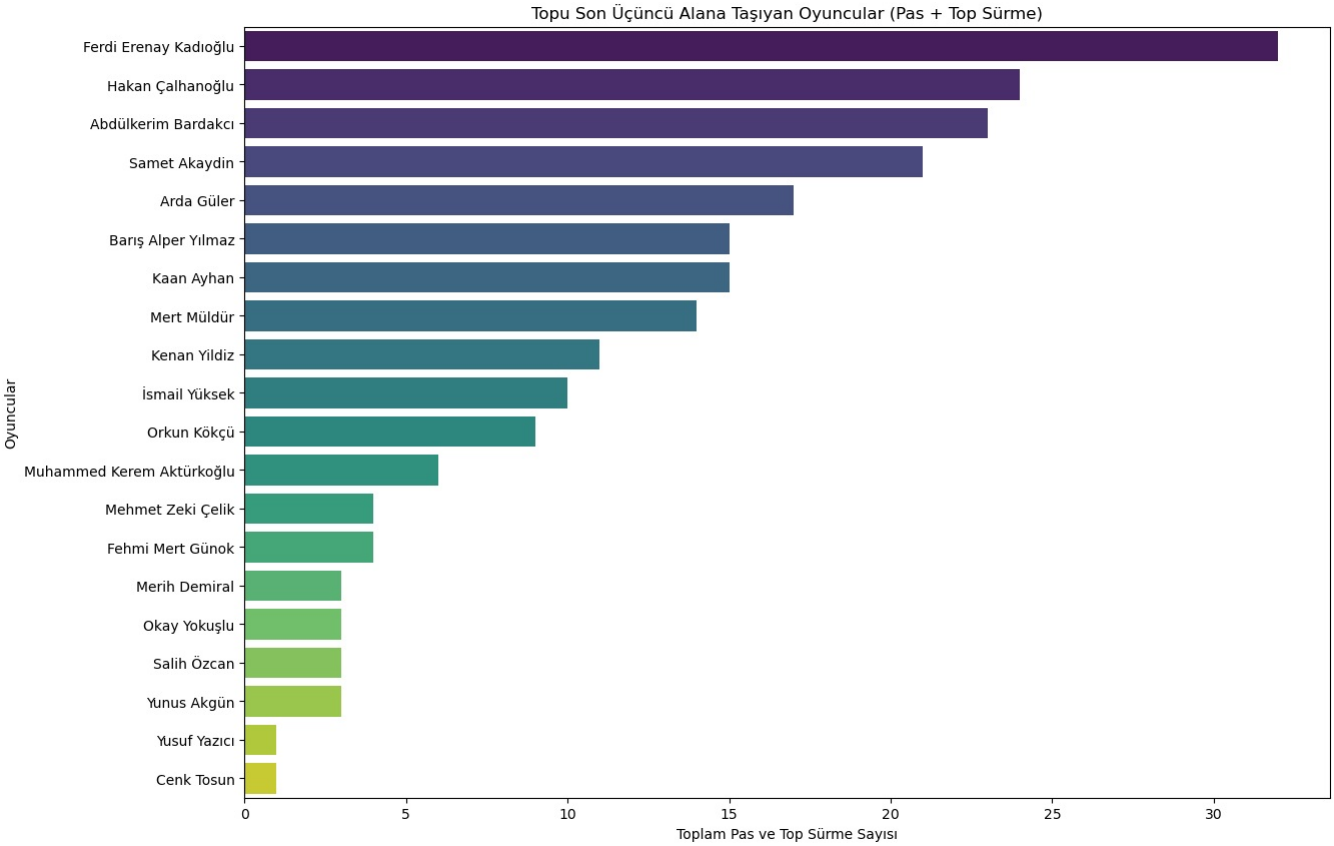
In [55]:
```
progression_df
```

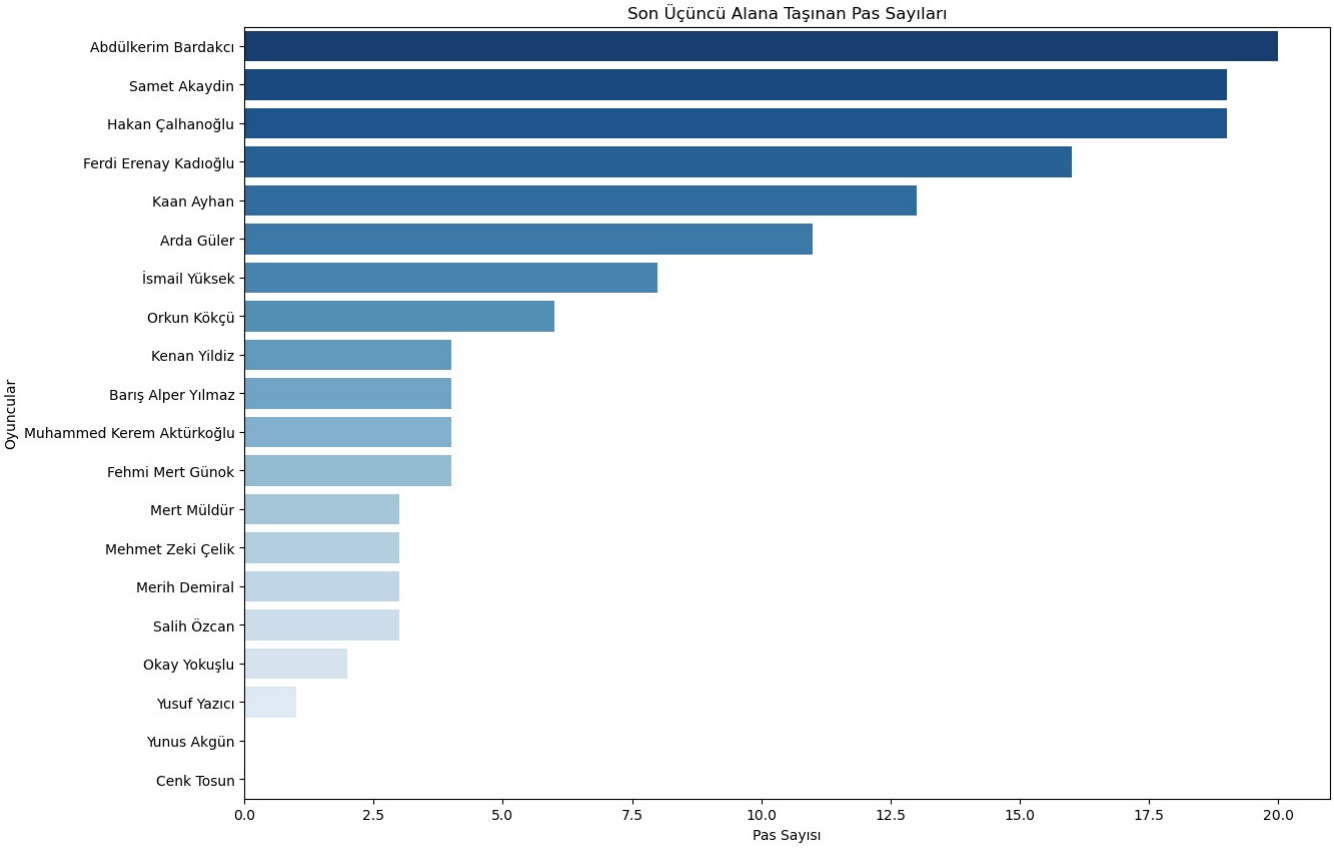
Out[55]:

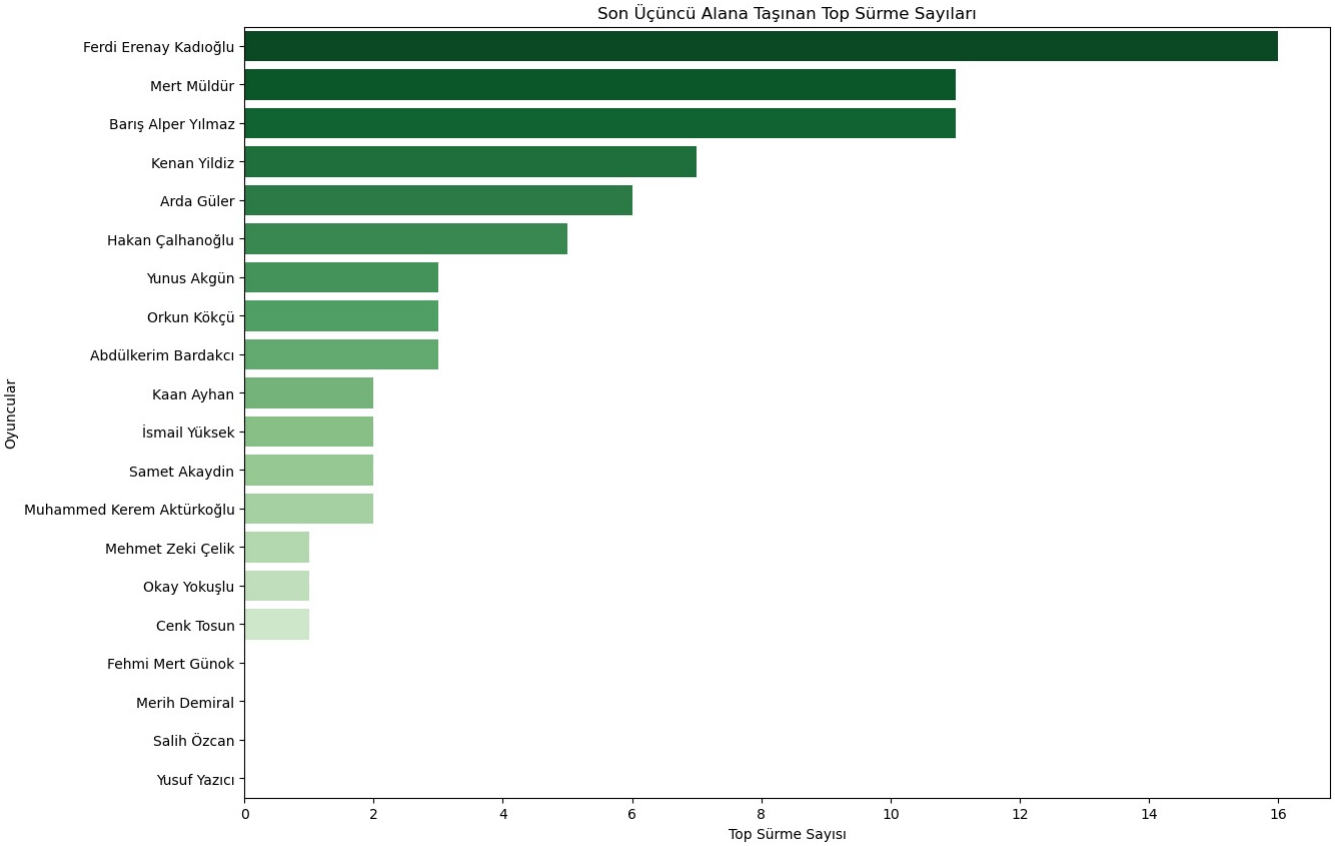
	player	Passes	Carries	total
4	Ferdi Erenay Kadıoğlu	16.0	16.0	32.0
5	Hakan Çalhanoğlu	19.0	5.0	24.0
0	Abdülkerim Bardakcı	20.0	3.0	23.0
15	Samet Akaydin	19.0	2.0	21.0
1	Arda Güler	11.0	6.0	17.0
2	Barış Alper Yılmaz	4.0	11.0	15.0
6	Kaan Ayhan	13.0	2.0	15.0
10	Mert Müldür	3.0	11.0	14.0
7	Kenan Yildiz	4.0	7.0	11.0
17	İsmail Yüksek	8.0	2.0	10.0
13	Orkun Kökçü	6.0	3.0	9.0
11	Muhammed Kerem Aktürkoğlu	4.0	2.0	6.0
8	Mehmet Zeki Çelik	3.0	1.0	4.0
3	Fehmi Mert Günok	4.0	0.0	4.0
9	Merih Demiral	3.0	0.0	3.0
12	Okay Yokuşlu	2.0	1.0	3.0
14	Salih Özcan	3.0	0.0	3.0
19	Yunus Akgün	0.0	3.0	3.0
16	Yusuf Yazıcı	1.0	0.0	1.0
18	Cenk Tosun	0.0	1.0	1.0

In [61]:

```
# Toplam Pas ve Top Sürme Sayısını Yönlendiren Bir Bar Grafiği
plt.figure(figsize=(14, 10))

progression_df_sorted = progression_df.sort_values(by='total', ascending=False)

sns.barplot(x='total', y='player', data=progression_df_sorted, palette='viridis')
plt.title('Topu Son Üçüncü Alana Taşıyan Oyuncular (Pas + Top Sürme)')
plt.xlabel('Toplam Pas ve Top Sürme Sayısı')
plt.ylabel('Oyuncular')
plt.show()


# Pas Sayıları
plt.figure(figsize=(14, 10))

progression_df_sorted_passes = progression_df.sort_values(by='Passes', ascending=False)

sns.barplot(x='Passes', y='player', data=progression_df_sorted_passes, palette='Blues_r')
plt.title('Son Üçüncü Alana Taşınan Pas Sayıları')
plt.xlabel('Pas Sayısı')
plt.ylabel('Oyuncular')
plt.show()

# Top Sürme Sayıları
plt.figure(figsize=(14, 10))

progression_df_sorted_carries = progression_df.sort_values(by='Carries', ascending=False)

sns.barplot(x='Carries', y='player', data=progression_df_sorted_carries, palette='Greens_r')
plt.title('Son Üçüncü Alana Taşınan Top Sürme Sayıları')
plt.xlabel('Top Sürme Sayısı')
plt.ylabel('Oyuncular')
plt.show()
```


Topu Son Üçüncü Alana Taşıyan Oyuncular (Pas + Top Sürme)
Ferdi Erenay Kadıoğlu
Hakan Çalhanoğlu
Abdülkerim Bardakcı
Samet Akaydin
Arda Güler
Barış Alper Yılmaz
Kaan Ayhan
Mert Müldür
Kenan Yildiz
İsmail Yüksek
Orkun Kökçü
Muhammed Kerem Aktürkoğlu
Mehmet Zeki Çelik
Fehmi Mert Günok
Merih Demiral
Okay Yokuşlu
Salih Özcan
Yunus Akgün
Yusuf Yazıcı
Cenk Tosun
Oyuncular
0
5
10
15
20
25
30
Toplam Pas ve Top Sürme Sayısı


Son Üçüncü Alana Taşınan Pas Sayıları
Abdülkerim Bardakcı
Samet Akaydin
Hakan Çalhanoğlu
Ferdi Erenay Kadıoğlu
Kaan Ayhan
Arda Güler
İsmail Yüksek
Orkun Kökçü
Kenan Yildiz
Barış Alper Yılmaz
Muhammed Kerem Aktürkoğlu
Fehmi Mert Günok
Mert Müldür
Mehmet Zeki Çelik
Merih Demiral
Salih Özcan
Okay Yokuşlu
Yusuf Yazıcı
Yunus Akgün
Cenk Tosun
Oyuncular
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5
20.0
Pas Sayısı

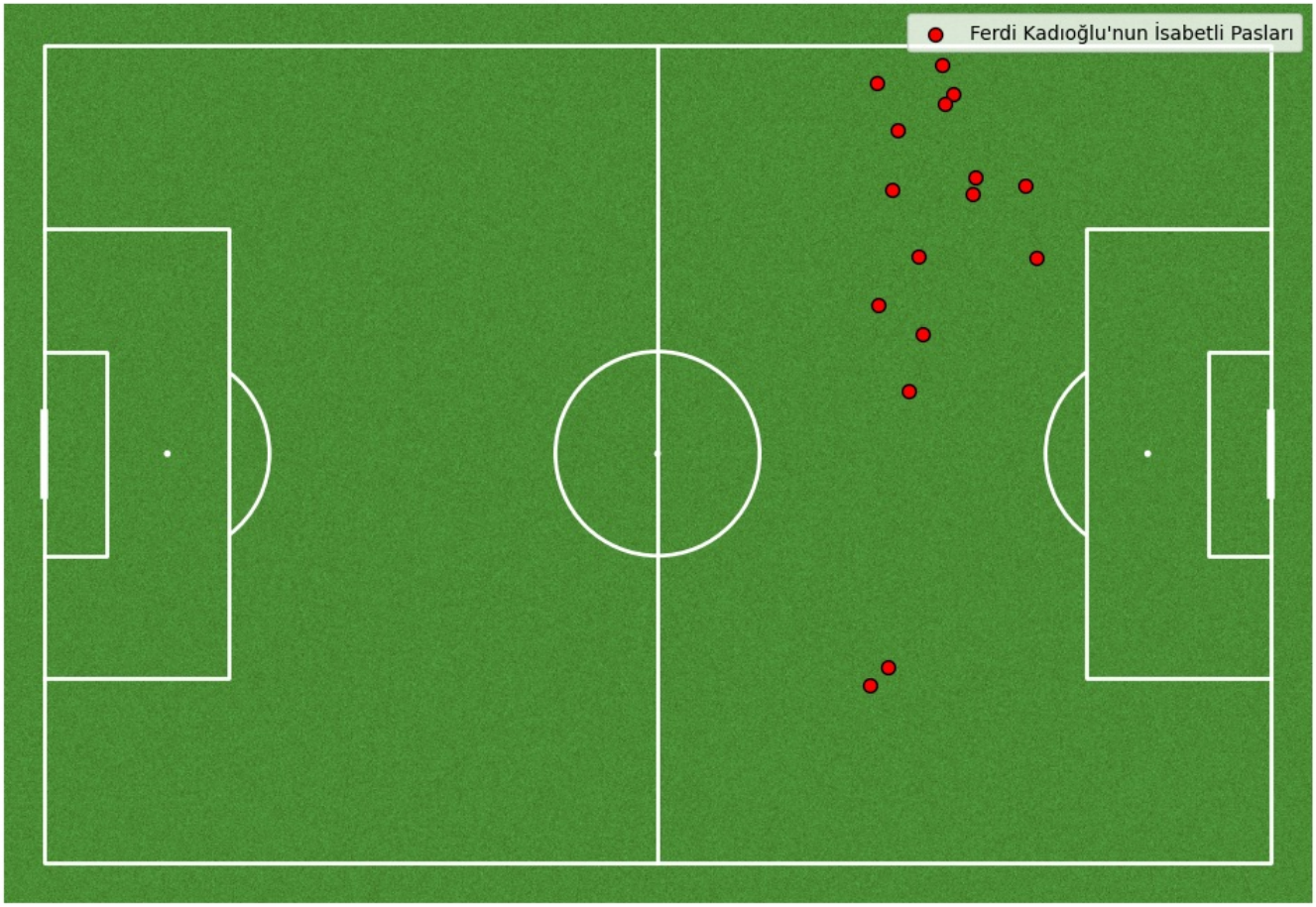


In [63]:
```
events_all_df.columns
```

Out[63]:
```
Index(['50_50', 'bad_behaviour_card', 'ball_receipt_outcome',
       'ball_recovery_recovery_failure', 'block_deflection',
       'block_save_block', 'carry_end_location', 'clearance_aerial_won',
       'clearance_body_part', 'clearance_head',
       ...
       'shot_deflected', 'ball_recovery_offensive', 'pass_miscommunication',
       'dribble_no_touch', 'x', 'y', 'pass_end_x', 'pass_end_y', 'carry_end_x',
       'carry_end_y'],
      dtype='object', length=110)
```

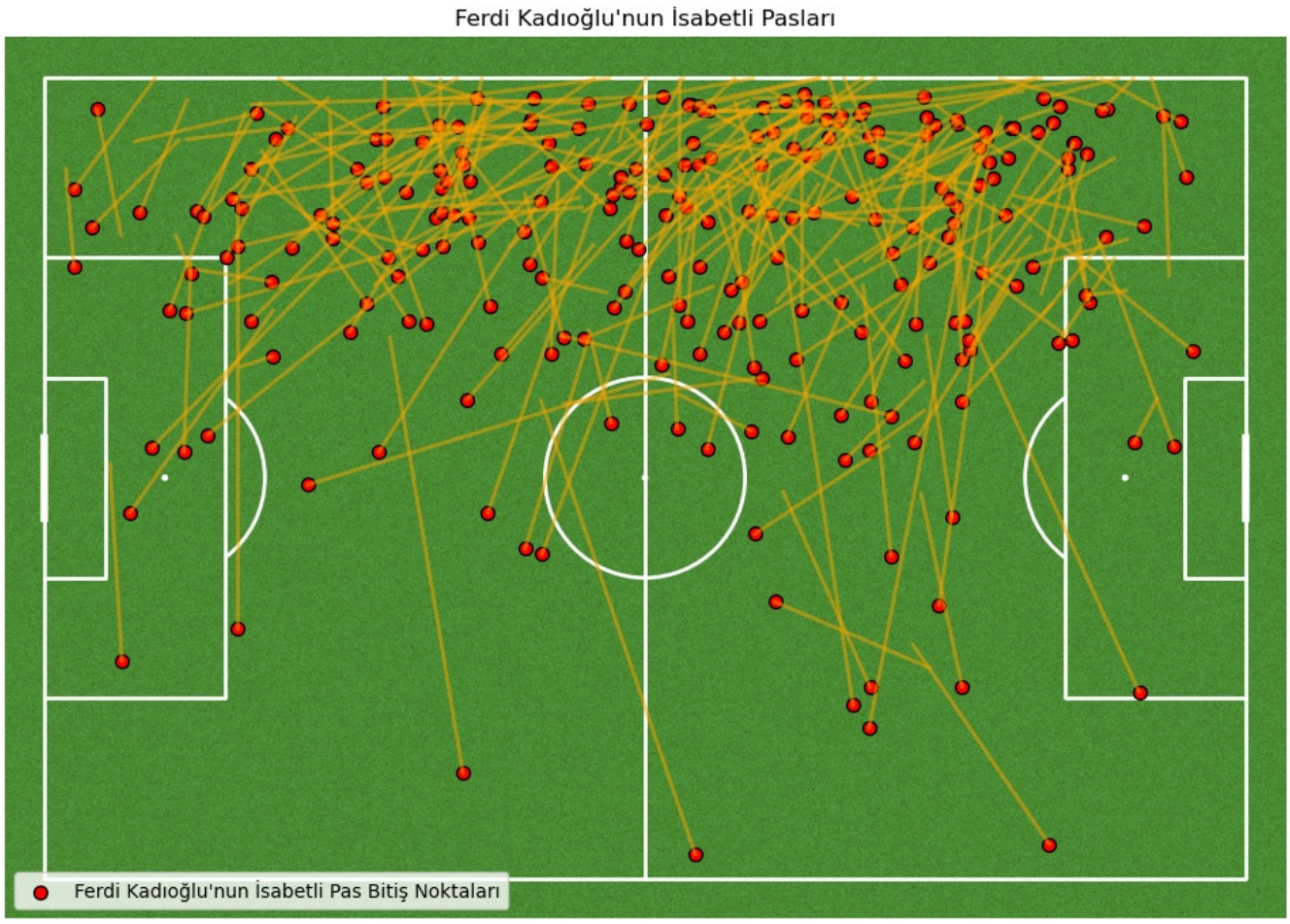
In [69]:
```
# Futbol sahasını oluşturma
pitch = Pitch(pitch_type='statsbomb', pitch_color='grass', line_color='white')
fig, ax = pitch.draw(figsize=(15, 7))

# Ferdi'nin isabetli paslarını filtreleme
ferdi_passes = events_all_df[
    (events_all_df.player == 'Ferdi Erenay Kadıoğlu') &
    (events_all_df.type == "Pass") &
    (events_all_df.pass_outcome.isna()) &
    (events_all_df.x < 80) &
    (events_all_df.pass_end_x > 80)
]

# ferdi'in isabetli paslarını sahada gösterme
pitch.scatter(
    x=ferdi_passes.pass_end_x,
    y=ferdi_passes.pass_end_y,
    c='red',
    label='Ferdi Kadıoğlu\'nun İsabetli Pasları',
    ax=ax,
    s=50,
    edgecolor='black'
)

plt.title('Ferdi Kadıoğlu\'nun İsabetli Pasları(3.Bölge)')
plt.legend()
plt.show()
```

Ferdi Kadıoğlu'nun İsabetli Pasları(3.Bölge)



In [71]:

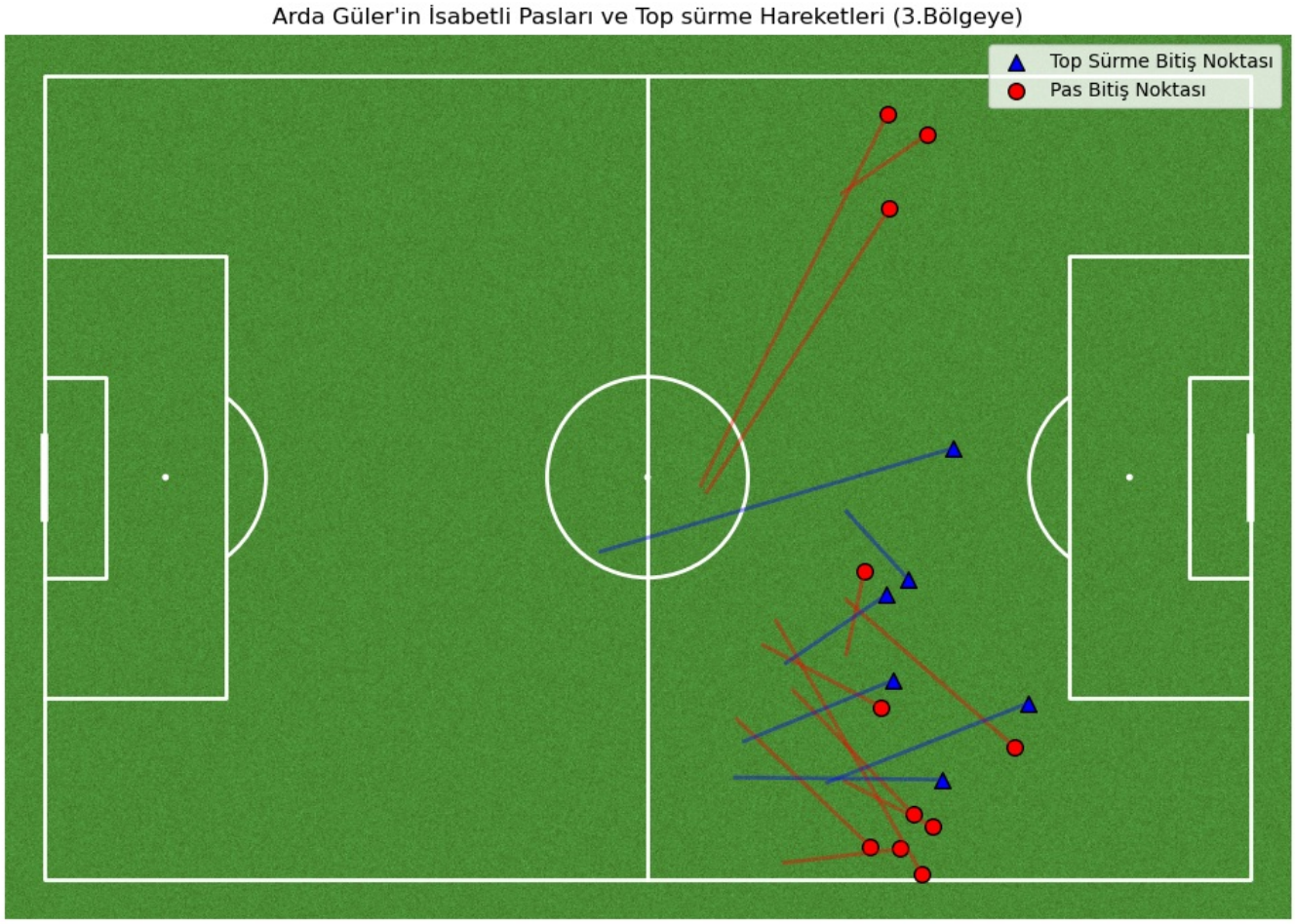
```
# Futbol sahasını oluşturma
pitch = Pitch(pitch_type='statsbomb', pitch_color='grass', line_color='white')
fig, ax = pitch.draw(figsize=(15, 7))

# Ferdi'nin isabetli paslarını filtreleme
ferdi_passes = events_all_df[
    (events_all_df.player == 'Ferdi Erenay Kadıoğlu') &
    (events_all_df.type == "Pass") &
    (events_all_df.pass_outcome.isna())
]

# pasların başlangıç ve bitiş noktalarını çizme
for _, row in ferdi_passes.iterrows():
    ax.plot([row['x'], row['pass_end_x']], [row['y'], row['pass_end_y']], color='orange', linestyle='-', linewi

# pasların bitiş noktalarını işaretleme
pitch.scatter(
    x=ferdi_passes.pass_end_x,
    y=ferdi_passes.pass_end_y,
    c='red',
    label='Ferdi Kadıoğlu\'nun İsabetli Pas Bitiş Noktaları',
    ax=ax,
    s=50,
    edgecolor='black'
)

plt.title('Ferdi Kadıoğlu\'nun İsabetli Pasları')
plt.legend()
plt.show()
```



In [73]:

```
# Futbol sahasını oluşturma
pitch = Pitch(pitch_type='statsbomb', pitch_color='grass', line_color='white')
fig, ax = pitch.draw(figsize=(15, 7))

# Arda Güler'in isabetli paslarını filtreleme
hakancalh_passes = events_all_df[
    (events_all_df.player == 'Arda Güler') &
    (events_all_df.type == "Pass") &
    (events_all_df.pass_outcome.isna()) &
    (events_all_df.x < 80) &
    (events_all_df.pass_end_x > 80)
]

# Arda Güler'in carry hareketlerini filtreleme
hakancalh_carries = events_all_df[
    (events_all_df.player == 'Arda Güler') &
    (events_all_df.type == "Carry") &
    (events_all_df.x < 80) &
    (events_all_df.carry_end_x > 80)
]

# Pasların başlangıç ve bitiş noktalarını çizme
for _, row in hakancalh_passes.iterrows():
    ax.plot([row['x'], row['pass_end_x']], [row['y'], row['pass_end_y']], color='red', linestyle='-', linewidth

# Carry hareketlerinin başlangıç ve bitiş noktalarını çizme
for _, row in hakancalh_carries.iterrows():
    ax.plot([row['x'], row['carry_end_x']], [row['y'], row['carry_end_y']], color='blue', linestyle='-', linewi
    # Carry hareketlerinin bitiş noktalarına mavi üçgen koyma
    ax.scatter(row['carry_end_x'], row['carry_end_y'], color='blue', s=70, edgecolor='black', marker='^', label

# Pasların bitiş noktalarını işaretleme
ax.scatter(
    x=hakancalh_passes.pass_end_x,
    y=hakancalh_passes.pass_end_y,
    color='red',
    s=70,
    edgecolor='black',
    marker='o',
    label='Pas Bitiş Noktası'
)

# İşaretlerin tekrar etmesini engelleme
handles, labels = ax.get_legend_handles_labels()
by_label = dict(zip(labels, handles))
ax.legend(by_label.values(), by_label.keys())

plt.title('Arda Güler\'in İsabetli Pasları ve Top sürme Hareketleri (3.Bölgeye)')
```

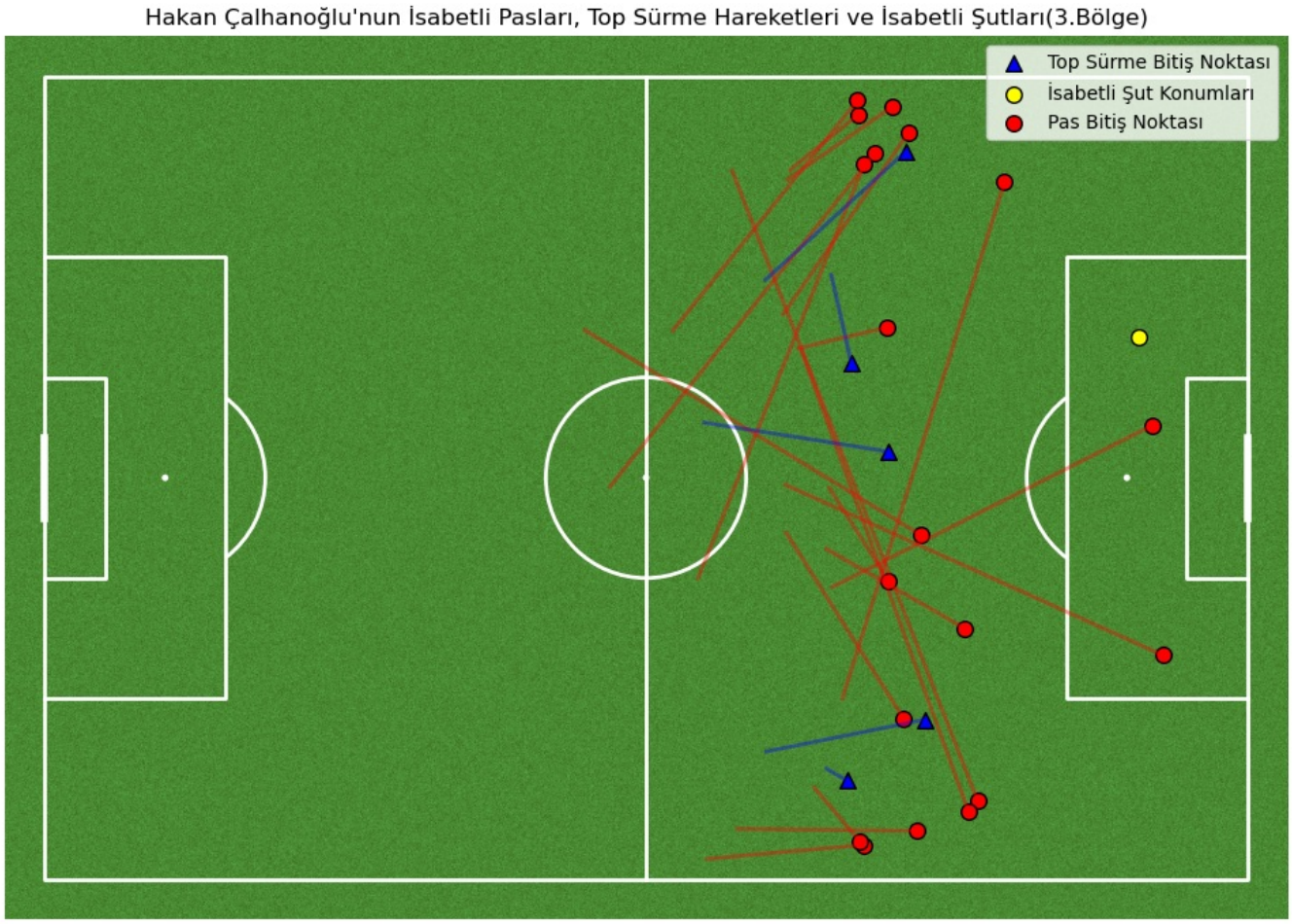
```
plt.show()
```



In [75]:

```
pitch = Pitch(pitch_type='statsbomb', pitch_color='grass', line_color='white')
fig, ax = pitch.draw(figsize=(15, 7))

# Hakan Çalhanoğlu'nun isabetli paslarını filtreleme
hakancalh_passes = events_all_df[
    (events_all_df.player == 'Hakan Çalhanoğlu') &
    (events_all_df.type == "Pass") &
    (events_all_df.pass_outcome.isna()) &
    (events_all_df.x < 80) &
    (events_all_df.pass_end_x > 80)
]

# Hakan Çalhanoğlu'nun top sürme hareketlerini filtreleme
hakancalh_carries = events_all_df[
    (events_all_df.player == 'Hakan Çalhanoğlu') &
    (events_all_df.type == "Carry") &
    (events_all_df.x < 80) &
    (events_all_df.carry_end_x > 80)
]

# Hakan Çalhanoğlu'nun isabetli şutlarını filtreleme
hakancalh_shots = events_all_df[
    (events_all_df.player == 'Hakan Çalhanoğlu') &
    (events_all_df.type == "Shot") &
    (events_all_df.shot_outcome == "Goal")
   # Şutların vurulduğu konumlar (genellikle orta saha ve ileri bölgede)
]

# Pasların başlangıç ve bitiş noktalarını çizme
for _, row in hakancalh_passes.iterrows():
    ax.plot([row['x'], row['pass_end_x']], [row['y'], row['pass_end_y']], color='red', linestyle='-', linewidth

# Carry hareketlerinin başlangıç ve bitiş noktalarını çizme
for _, row in hakancalh_carries.iterrows():
    ax.plot([row['x'], row['carry_end_x']], [row['y'], row['carry_end_y']], color='blue', linestyle='-', linewi
    # Carry hareketlerinin bitiş noktalarına mavi üçgen koyma
    ax.scatter(row['carry_end_x'], row['carry_end_y'], color='blue', s=70, edgecolor='black', marker='^', label

# İsabetli şutların vurulduğu konumları işaretleme
ax.scatter(
    x=hakancalh_shots.x,
    y=hakancalh_shots.y,
    color='yellow',
    s=70,
    edgecolor='black',
    marker='o',
```

```
    label='İsabetli Şut Konumları'
)

# Pasların bitiş noktalarını işaretleme
ax.scatter(
    x=hakancalh_passes.pass_end_x,
    y=hakancalh_passes.pass_end_y,
    color='red',
    s=70,
    edgecolor='black',
    marker='o',
    label='Pas Bitiş Noktası'
)

# İşaretlerin tekrar etmesini engelleme
handles, labels = ax.get_legend_handles_labels()
by_label = dict(zip(labels, handles))
ax.legend(by_label.values(), by_label.keys())

plt.title('Hakan Çalhanoğlu\'nun İsabetli Pasları, Top Sürme Hareketleri ve İsabetli Şutları(3.Bölge)')
plt.show()
```



In [77]:
```
events_all_df.columns[60:]
```

Out[77]:
```
Index(['pass_type', 'period', 'play_pattern', 'player', 'player_id',
       'position', 'possession', 'possession_team', 'possession_team_id',
       'related_events', 'second', 'shot_aerial_won', 'shot_body_part',
       'shot_end_location', 'shot_first_time', 'shot_freeze_frame',
       'shot_key_pass_id', 'shot_open_goal', 'shot_outcome',
       'shot_saved_to_post', 'shot_statsbomb_xg', 'shot_technique',
       'shot_type', 'substitution_outcome', 'substitution_outcome_id',
       'substitution_replacement', 'substitution_replacement_id', 'tactics',
       'team', 'team_id', 'timestamp', 'type', 'under_pressure',
       'block_offensive', 'dribble_overrun', 'pass_outswinging',
       'shot_one_on_one', 'clearance_other', 'pass_cut_back', 'pass_no_touch',
       'shot_deflected', 'ball_recovery_offensive', 'pass_miscommunication',
       'dribble_no_touch', 'x', 'y', 'pass_end_x', 'pass_end_y', 'carry_end_x',
       'carry_end_y'],
      dtype='object')
```

In [79]:
```
events_all_df["type"].value_counts()
```

Out[79]:
```
type
Pass                5007
Ball Receipt*       4861
Carry               4059
Pressure            1297
Ball Recovery        448
Duel                 345
Clearance            192
Block                190
Goal Keeper          172
Dribble              169
Shot                 142
Foul Committed       117
Foul Won             112
Miscontrol           110
Dispossessed          95
Dribbled Past         84
Interception          78
Substitution          47
Injury Stoppage       33
50/50                 32
Half Start            20
Half End              20
Tactical Shift        18
Bad Behaviour         16
Referee Ball-Drop     11
Starting XI           10
Player On              8
Player Off             8
Shield                 4
Error                  3
Own Goal Against       2
Own Goal For           2
Offside                1
Name: count, dtype: int64
```

In [81]:
```
# 'type' sütunundaki benzersiz değerleri görüntüleme
unique_types = events_all_df['type'].unique()
print("Benzersiz 'type' değerleri:", unique_types)

# 'shot_outcome' sütunundaki benzersiz değerleri görüntüleme
unique_shot_outcomes = events_all_df['shot_outcome'].unique()
print("Benzersiz 'shot_outcome' değerleri:", unique_shot_outcomes)
```

```
Benzersiz 'type' değerleri: ['Starting XI' 'Half Start' 'Pass' 'Ball Receipt*' 'Carry' 'Duel'
 'Clearance' 'Pressure' 'Block' 'Ball Recovery' 'Dribbled Past' 'Dribble'
 'Shot' 'Goal Keeper' 'Dispossessed' 'Interception' 'Miscontrol'
 'Foul Committed' 'Foul Won' '50/50' 'Injury Stoppage' 'Player Off'
 'Player On' 'Shield' 'Half End' 'Substitution' 'Tactical Shift'
 'Referee Ball-Drop' 'Own Goal Against' 'Own Goal For' 'Bad Behaviour'
 'Error' 'Offside']
Benzersiz 'shot_outcome' değerleri: [nan 'Off T' 'Blocked' 'Goal' 'Saved to Post' 'Saved' 'Wayward' 'Post']
```

nan: Eksik veya bilinmeyen sonuçlar.

Off T: Kaleyi bulmayan şutlar.

Blocked: Rakip oyuncu veya savunma oyuncusu tarafından engellenmiş şutlar.

Goal: Gol olan şutlar.

Saved to Post: Kaleci tarafından direğe çarpan ama gol olamayan şutlar.

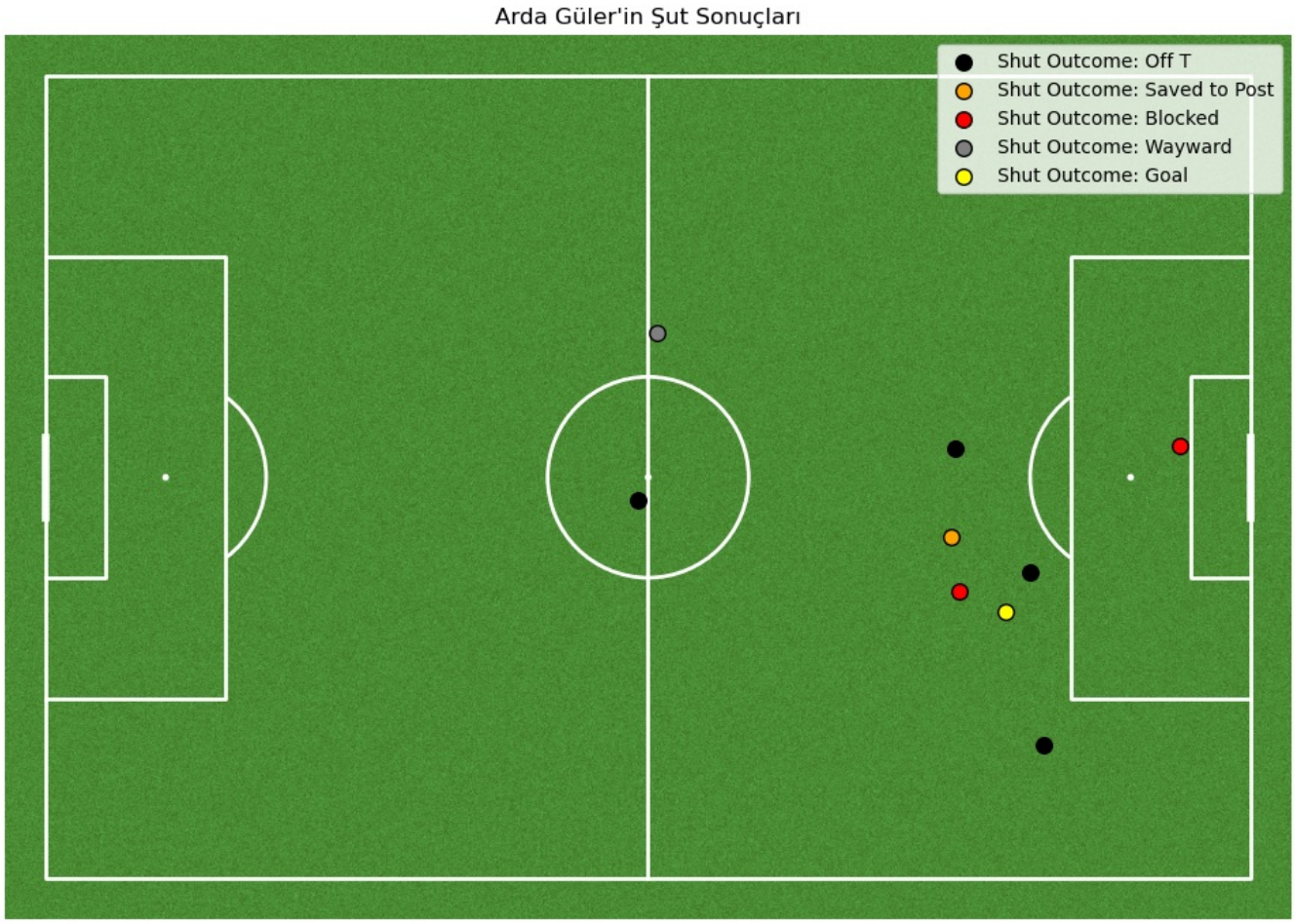
Saved: Kaleci tarafından kurtarılan şutlar.

Wayward: Kaleye isabet etmeyen ve hedeften uzaklaşan şutlar.

Post: Direğe çarpan şutlar

In [83]:
```
import matplotlib.pyplot as plt
from matplotlib.patches import Arc

pitch = Pitch(pitch_type='statsbomb', pitch_color='grass', line_color='white')
fig, ax = pitch.draw(figsize=(15, 7))

# Arda Güler'in şutlarını filtreleme
hakancalh_shots = events_all_df[
    (events_all_df.player == 'Arda Güler') &
    (events_all_df.type == "Shot")
]

# Şutların sonuçlarına göre renkler ve işaretler tanımlama
colors = {
    'Goal': 'yellow',
    'Blocked': 'red',
    'Saved to Post': 'orange',
```

```
    'Saved': 'blue',
    'Wayward': 'gray',
    'Post': 'green',
    'nan': 'black'  # Eksik veri için siyah
}

# Şutları işaretleme
for _, row in hakancalh_shots.iterrows():
    outcome = row['shot_outcome']
    color = colors.get(outcome, 'black')  # Bilinmeyen durumlar için siyah
    ax.scatter(
        x=row['x'],
        y=row['y'],
        color=color,
        s=70,
        edgecolor='black',
        marker='o',
        label=f'Shut Outcome: {outcome}'
    )

# Şutların üzerine açıklama ekleme
handles, labels = ax.get_legend_handles_labels()
by_label = dict(zip(labels, handles))
ax.legend(by_label.values(), by_label.keys())

plt.title('Arda Güler\'in Şut Sonuçları')
plt.show()
```



In [85]:
```
events_all_df["type"].value_counts()
```

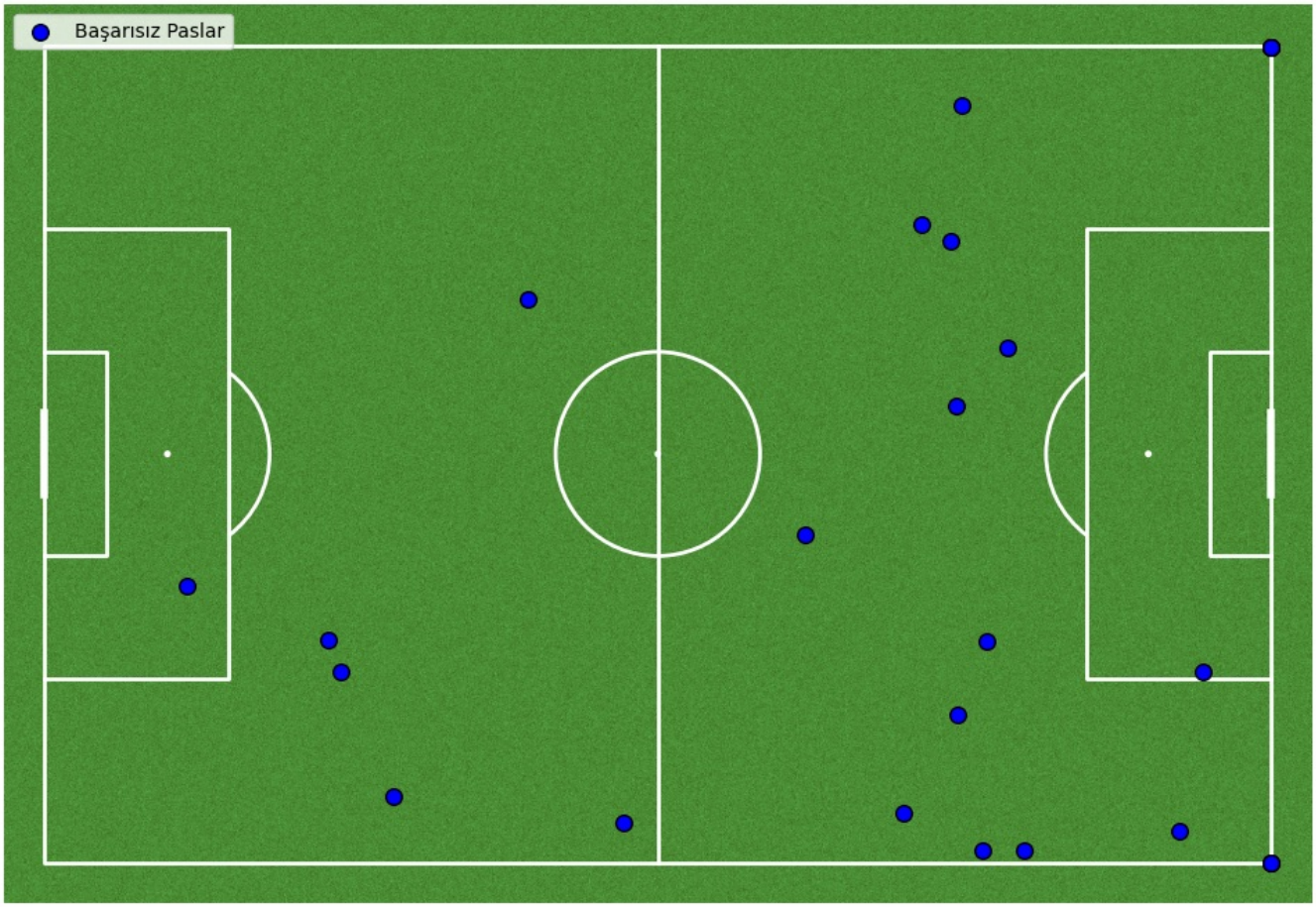
Out[85]:
```
type
Pass                5007
Ball Receipt*       4861
Carry               4059
Pressure            1297
Ball Recovery        448
Duel                 345
Clearance            192
Block                190
Goal Keeper          172
Dribble              169
Shot                 142
Foul Committed       117
Foul Won             112
Miscontrol           110
Dispossessed          95
Dribbled Past         84
Interception          78
Substitution          47
Injury Stoppage       33
50/50                 32
Half Start            20
Half End              20
Tactical Shift        18
Bad Behaviour         16
Referee Ball-Drop     11
Starting XI           10
Player On              8
Player Off             8
Shield                 4
Error                  3
Own Goal Against       2
Own Goal For           2
Offside                1
Name: count, dtype: int64
```

In [87]:
```
events_all_df['shot_outcome'].value_counts()
```

Out[87]:
```
shot_outcome
Off T            43
Blocked          42
Saved            29
Goal             14
Wayward          11
Post              2
Saved to Post     1
Name: count, dtype: int64
```

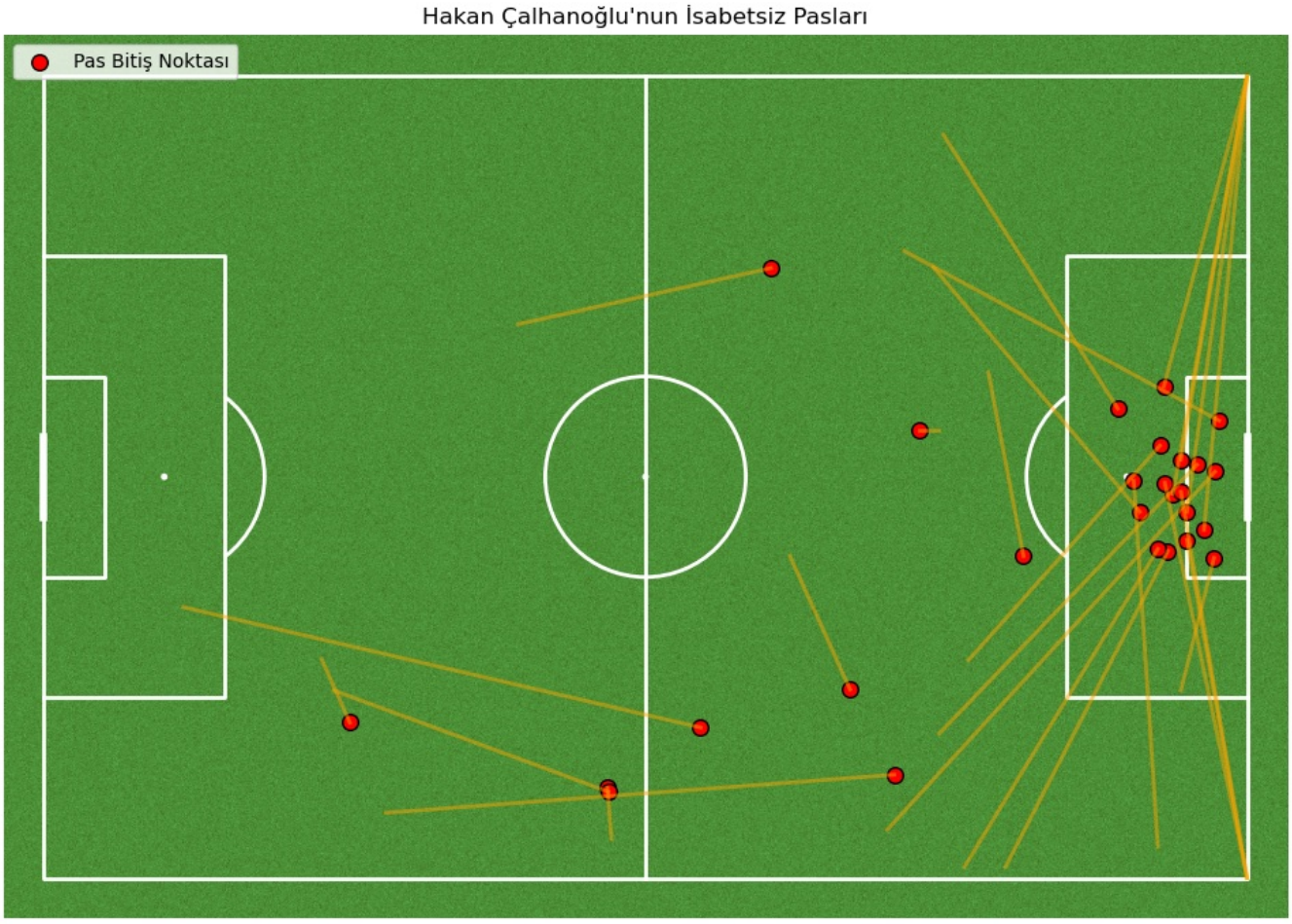
In [93]:
```
import matplotlib.pyplot as plt
from mplsoccer import Pitch

# Saha oluşturma
pitch = Pitch(pitch_type='statsbomb', pitch_color='grass', line_color='white')
fig, ax = pitch.draw(figsize=(15, 7))

# Hakan Çalhanoğlu'nun başarılı paslarını filtreleme
abdulkadir_passes = events_all_df[
    (events_all_df.player == 'Hakan Çalhanoğlu') &
    (events_all_df.type == "Pass") &
    (events_all_df.pass_outcome == 'Incomplete')
]

# Başarılı pasları işaretleme
ax.scatter(
    x=abdulkadir_passes['x'],
    y=abdulkadir_passes['y'],
    color='blue',
    s=70,
    edgecolor='black',
    marker='o',
    label='Başarısız Paslar'
)

handles, labels = ax.get_legend_handles_labels()
by_label = dict(zip(labels, handles))
ax.legend(by_label.values(), by_label.keys())
plt.title('Hakan Çalhanoğlu\'nun Başarısız Pasların Konumları')
plt.show()
```

Hakan Çalhanoğlu'nun Başarısız Pasların Konumları



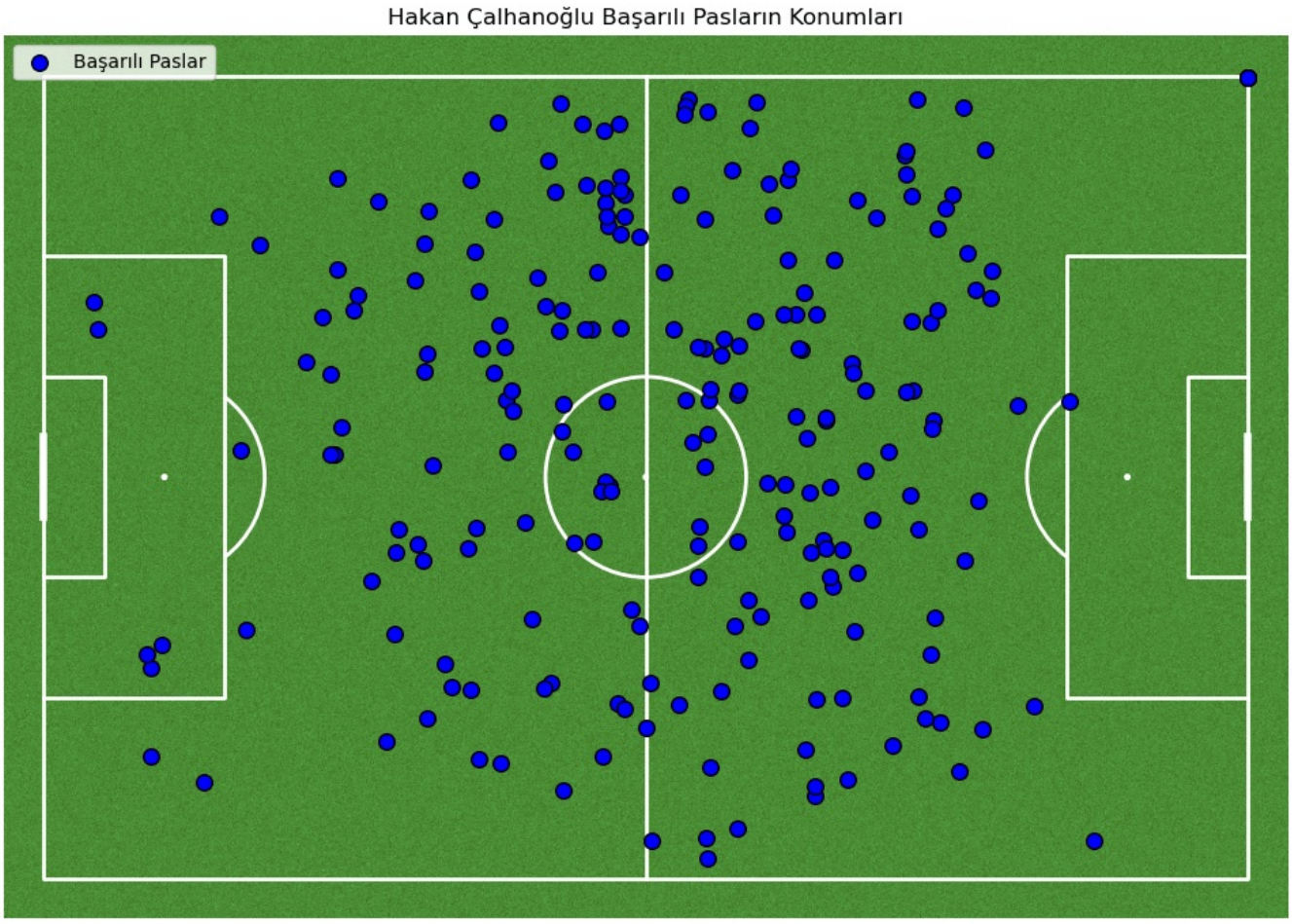
In [97]:

```
pitch = Pitch(pitch_type='statsbomb', pitch_color='grass', line_color='white')
fig, ax = pitch.draw(figsize=(15, 7))

# Hakan Çalhanoğlu'nun isabetli paslarını filtreleme
hakancalh_passes = events_all_df[
    (events_all_df.player == 'Hakan Çalhanoğlu') &
    (events_all_df.type == "Pass") &
    (events_all_df.pass_outcome == 'Incomplete')
]


# Pasların başlangıç ve bitiş noktalarını çizme
for _, row in hakancalh_passes.iterrows():
    ax.plot([row['x'], row['pass_end_x']], [row['y'], row['pass_end_y']], color='orange', linestyle='-', linewi



# Pasların bitiş noktalarını işaretleme
ax.scatter(
    x=hakancalh_passes.pass_end_x,
    y=hakancalh_passes.pass_end_y,
    color='red',
    s=70,
    edgecolor='black',
    marker='o',
    label='Pas Bitiş Noktası'
)

# İşaretlerin tekrar etmesini engelleme
handles, labels = ax.get_legend_handles_labels()
by_label = dict(zip(labels, handles))
ax.legend(by_label.values(), by_label.keys())

plt.title('Hakan Çalhanoğlu\'nun İsabetsiz Pasları')
plt.show()
```



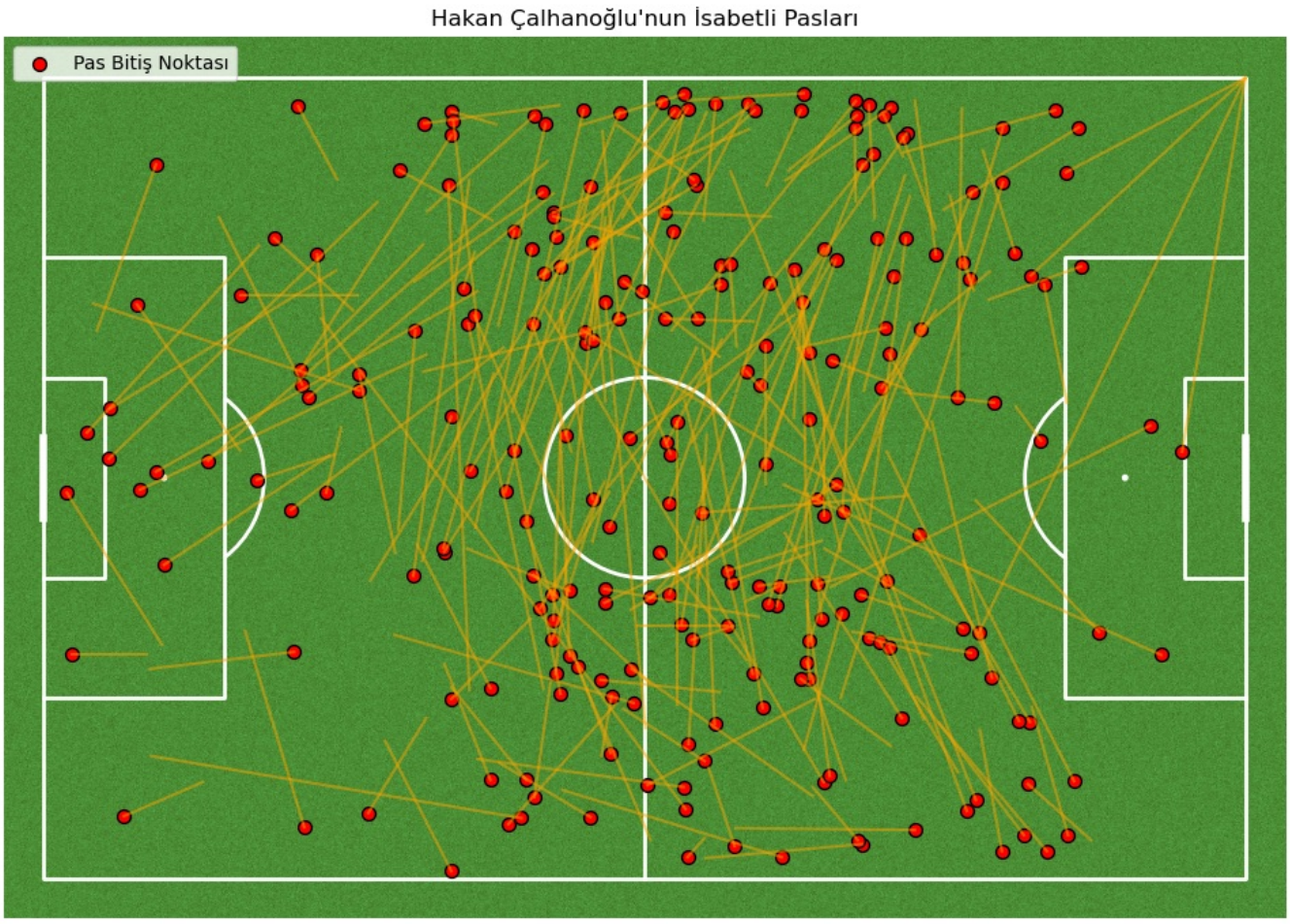
In [99]:

```
pitch = Pitch(pitch_type='statsbomb', pitch_color='grass', line_color='white')
fig, ax = pitch.draw(figsize=(15, 7))

abdulkadir_passes = events_all_df[
    (events_all_df.player == 'Hakan Çalhanoğlu') &
    (events_all_df.type == "Pass") &
    (events_all_df.pass_outcome.isna())
]

# Başarılı pasları işaretleme
ax.scatter(
    x=abdulkadir_passes['x'],
    y=abdulkadir_passes['y'],
    color='blue',
    s=70,
    edgecolor='black',
    marker='o',
    label='Başarılı Paslar'
)

handles, labels = ax.get_legend_handles_labels()
by_label = dict(zip(labels, handles))
ax.legend(by_label.values(), by_label.keys())
plt.title('Hakan Çalhanoğlu Başarılı Pasların Konumları')
plt.show()
```



In [101…

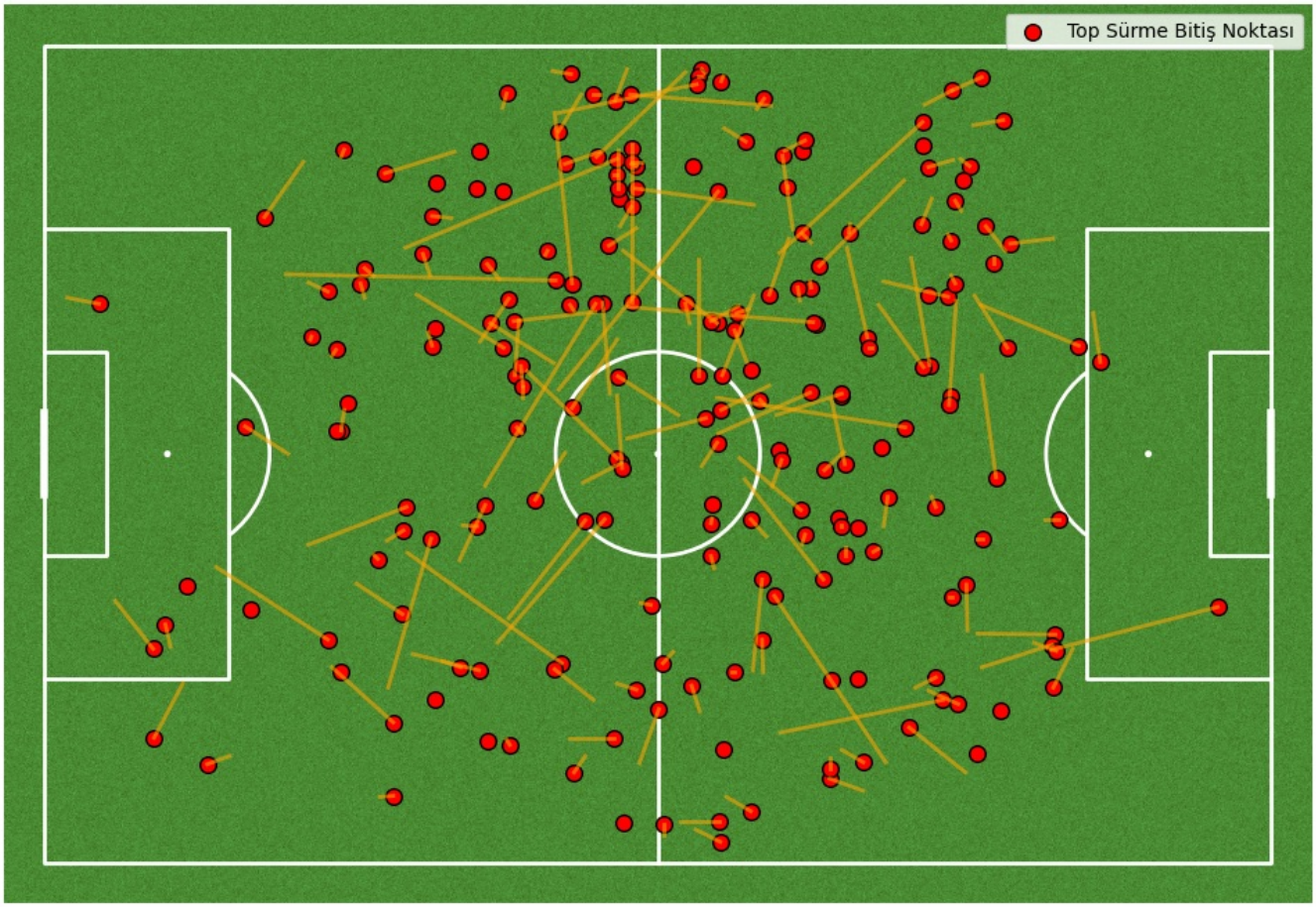
```
pitch = Pitch(pitch_type='statsbomb', pitch_color='grass', line_color='white')
fig, ax = pitch.draw(figsize=(15, 7))

# Hakan Çalhanoğlu'nun isabetli paslarını filtreleme
hakancalh_passes = events_all_df[
    (events_all_df.player == 'Hakan Çalhanoğlu') &
    (events_all_df.type == "Pass") &
    (events_all_df.pass_outcome.isna())

]


# Pasların başlangıç ve bitiş noktalarını çizme
for _, row in hakancalh_passes.iterrows():
    ax.plot([row['x'], row['pass_end_x']], [row['y'], row['pass_end_y']], color='orange', linestyle='-', linewi



# Pasların bitiş noktalarını işaretleme
ax.scatter(
    x=hakancalh_passes.pass_end_x,
    y=hakancalh_passes.pass_end_y,
    color='red',
    s=50,
    edgecolor='black',
    marker='o',
    label='Pas Bitiş Noktası'
)

# İşaretlerin tekrar etmesini engelleme
handles, labels = ax.get_legend_handles_labels()
by_label = dict(zip(labels, handles))
ax.legend(by_label.values(), by_label.keys())

# Görselleştirme ayarları
plt.title('Hakan Çalhanoğlu\'nun İsabetli Pasları')
plt.show()
```



In [103…

```
pitch = Pitch(pitch_type='statsbomb', pitch_color='grass', line_color='white')
fig, ax = pitch.draw(figsize=(15, 7))

# Belirli bir oyuncunun carry hareketlerini filtreleme
player_carries = events_all_df[
    (events_all_df['player'] == 'Hakan Çalhanoğlu') &
    (events_all_df['type'] == 'Carry')
]

# Carry hareketlerinin başlangıç ve bitiş noktalarını çizme
for _, row in player_carries.iterrows():
    ax.plot([row['x'], row['carry_end_x']], [row['y'], row['carry_end_y']], color='orange', linestyle='-', line

# Bitiş noktalarını işaretleme
ax.scatter(
    x=player_carries['carry_end_x'],
    y=player_carries['carry_end_y'],
    color='red',
    s=70,
    edgecolor='black',
    label='Top Sürme Bitiş Noktası'
)

# İşaretlerin tekrar etmesini engelleme
handles, labels = ax.get_legend_handles_labels()
by_label = dict(zip(labels, handles))
ax.legend(by_label.values(), by_label.keys())

# Görselleştirme ayarları
plt.title('Hakan Çalhanoğlu Top Sürme Haritası')
plt.show()
```

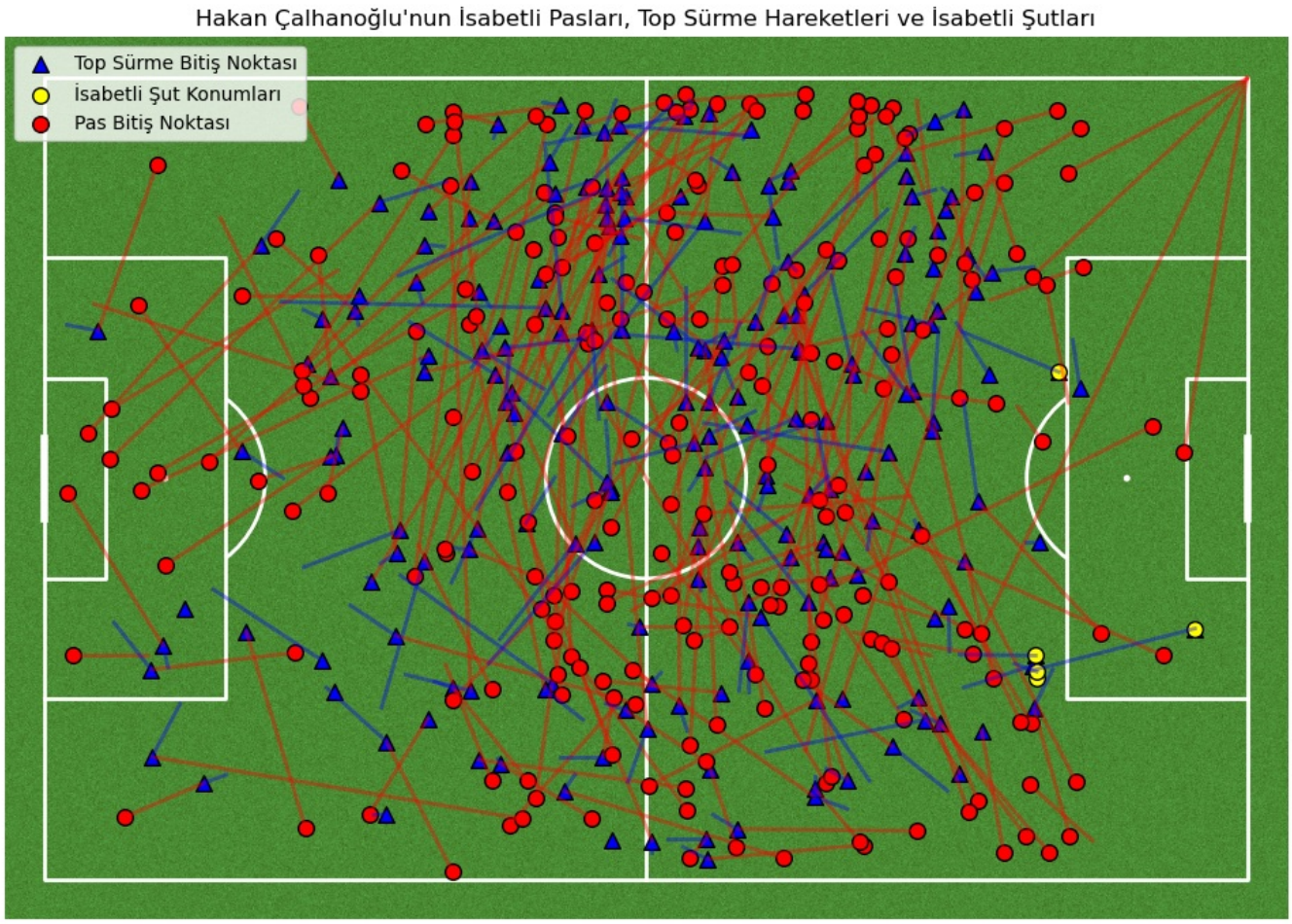
Hakan Çalhanoğlu Top Sürme Haritası



In [105…

```
pitch = Pitch(pitch_type='statsbomb', pitch_color='grass', line_color='white')
fig, ax = pitch.draw(figsize=(15, 7))

# Hakan Çalhanoğlu'nun isabetli paslarını filtreleme
hakancalh_passes = events_all_df[
    (events_all_df.player == 'Hakan Çalhanoğlu') &
    (events_all_df.type == "Pass") &
    (events_all_df.pass_outcome.isna())

]

# Hakan Çalhanoğlu'nun top sürme hareketlerini filtreleme
hakancalh_carries = events_all_df[
    (events_all_df.player == 'Hakan Çalhanoğlu') &
    (events_all_df.type == "Carry")
]

# Hakan Çalhanoğlu'nun isabetli şutlarını filtreleme
hakancalh_shots = events_all_df[
    (events_all_df.player == 'Hakan Çalhanoğlu') &
    (events_all_df.type == "Shot") &
    (events_all_df.shot_outcome == "Blocked")
   # Şutların vurulduğu konumlar (genellikle orta saha ve ileri bölgede)
]

# Pasların başlangıç ve bitiş noktalarını çizme
for _, row in hakancalh_passes.iterrows():
    ax.plot([row['x'], row['pass_end_x']], [row['y'], row['pass_end_y']], color='red', linestyle='-', linewidth

# Carry hareketlerinin başlangıç ve bitiş noktalarını çizme
for _, row in hakancalh_carries.iterrows():
    ax.plot([row['x'], row['carry_end_x']], [row['y'], row['carry_end_y']], color='blue', linestyle='-', linewi
    # Carry hareketlerinin bitiş noktalarına mavi üçgen koyma
    ax.scatter(row['carry_end_x'], row['carry_end_y'], color='blue', s=70, edgecolor='black', marker='^', label

# İsabetli şutların vurulduğu konumları işaretleme
ax.scatter(
    x=hakancalh_shots.x,
    y=hakancalh_shots.y,
    color='yellow',
    s=70,
    edgecolor='black',
    marker='o',
    label='İsabetli Şut Konumları'
)

# Pasların bitiş noktalarını işaretleme
ax.scatter(
```

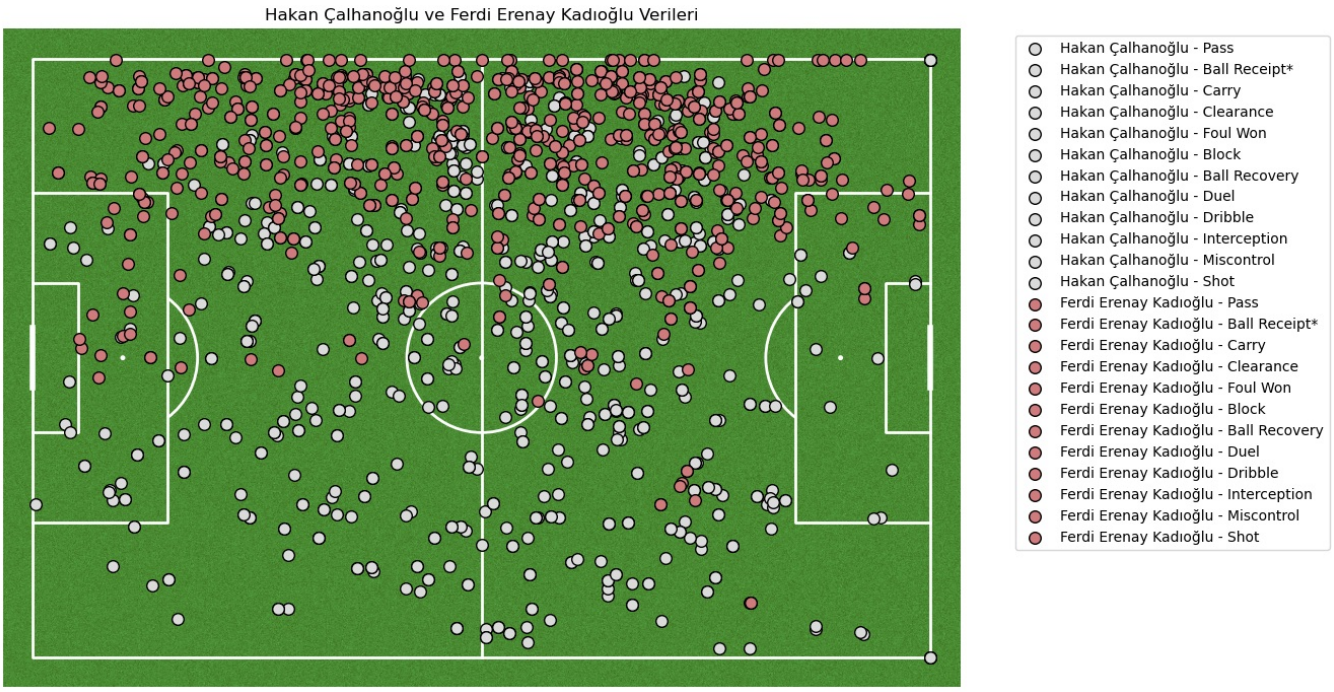
```
    x=hakancalh_passes.pass_end_x,
    y=hakancalh_passes.pass_end_y,
    color='red',
    s=70,
    edgecolor='black',
    marker='o',
    label='Pas Bitiş Noktası'
)

# İşaretlerin tekrar etmesini engelleme
handles, labels = ax.get_legend_handles_labels()
by_label = dict(zip(labels, handles))
ax.legend(by_label.values(), by_label.keys())

# Görselleştirme ayarları
plt.title('Hakan Çalhanoğlu\'nun İsabetli Pasları, Top Sürme Hareketleri ve İsabetli Şutları')
plt.show()
```



In [107…

```
# Oyuncular ve değişkenler
player1 = "Hakan Çalhanoğlu"
player2 = "Ferdi Erenay Kadıoğlu"
touches = ["Pass", "Ball Receipt*", "Carry", "Clearance", "Foul Won", "Block", "Ball Recovery", "Duel", "Dribbl

# Veri seti filtreleme
player1_df = events_all_df[(events_all_df['player'] == player1) & (events_all_df['type'].isin(touches))]
player2_df = events_all_df[(events_all_df['player'] == player2) & (events_all_df['type'].isin(touches))]

colour1 = "white"
colour2 = "#c3c3c3"
colour3 = "#e21017"
cmaplist = [colour1, colour2, colour3]
cmap = LinearSegmentedColormap.from_list("", cmaplist)

# saha olusturma
pitch = Pitch(pitch_type='statsbomb', pitch_color='grass', line_color='white')
fig, ax = pitch.draw(figsize=(15, 7))

# Hakan Çalhanoğlu
for event_type in touches:
    event_df = player1_df[player1_df['type'] == event_type]
    ax.scatter(
        x=event_df['x'],
        y=event_df['y'],
        color=cmap(0.3),  # Her bir olay türü için farklı renk tonları kullanılabilir
        s=70,
        edgecolor='black',
        label=f'{player1} - {event_type}'
    )
```

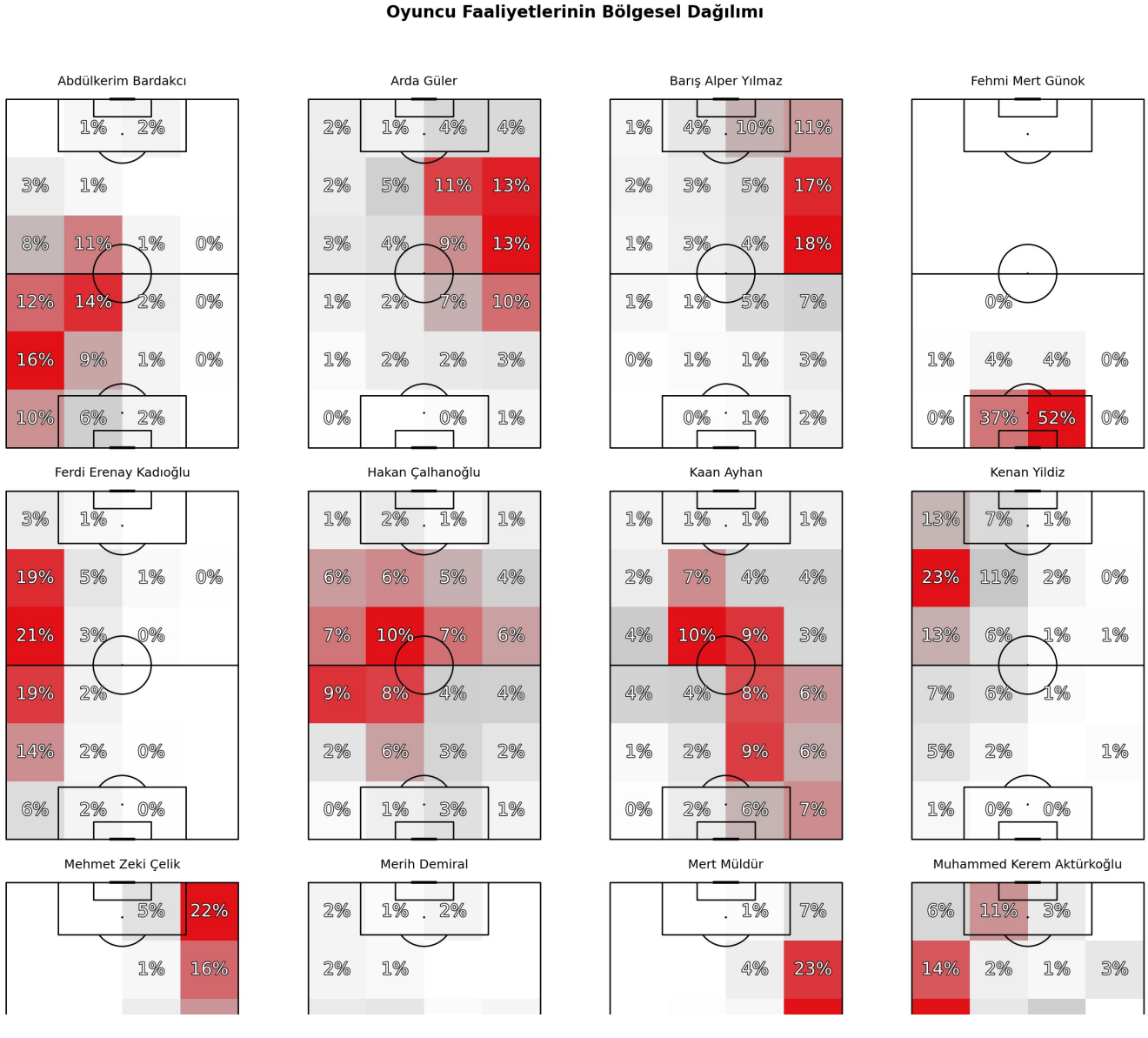
```
# Ferdi Erenay Kadıoğlu
for event_type in touches:
    event_df = player2_df[player2_df['type'] == event_type]
    ax.scatter(
        x=event_df['x'],
        y=event_df['y'],
        color=cmap(0.7),  # Her bir olay türü için farklı renk tonları kullanılabilir
        s=70,
        edgecolor='black',
        label=f'{player2} - {event_type}'
    )

# İşaretlerin tekrar etmesini engelleme
handles, labels = ax.get_legend_handles_labels()
by_label = dict(zip(labels, handles))
ax.legend(by_label.values(), by_label.keys(), bbox_to_anchor=(1.05, 1), loc='upper left')

plt.title(f'{player1} ve {player2} Verileri')
plt.show()
```

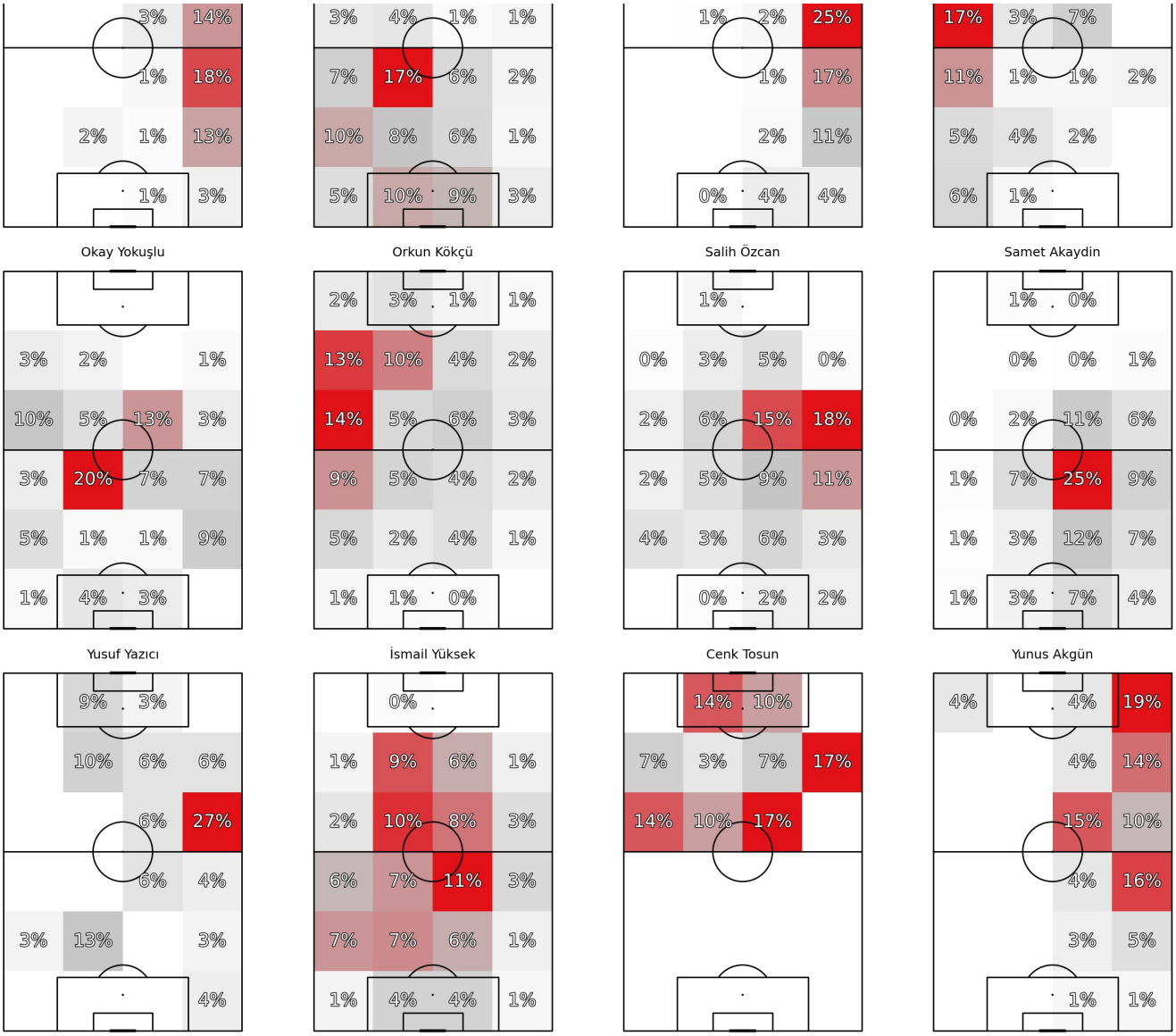


In [109…

```
colour1 = "white"
colour2 = "#c3c3c3"
colour3 = "#e21017"
cmaplist = [colour1, colour2, colour3]
cmap = LinearSegmentedColormap.from_list("", cmaplist)

# Path effect ayarları
path_eff = [path_effects.Stroke(linewidth=2, foreground="black"), path_effects.Normal()]

# saha olusturma
pitch = VerticalPitch(pitch_type="statsbomb", line_zorder=2, line_color="#000000", linewidth=2, half=False)

# Türkiye kadrosunu filtreleme
players = [
    'Abdülkerim Bardakcı', 'Arda Güler', 'Barış Alper Yılmaz', 'Fehmi Mert Günok', 'Ferdi Erenay Kadıoğlu',
    'Hakan Çalhanoğlu', 'Kaan Ayhan', 'Kenan Yildiz', 'Mehmet Zeki Çelik', 'Merih Demiral',
    'Mert Müldür', 'Muhammed Kerem Aktürkoğlu', 'Okay Yokuşlu', 'Orkun Kökçü', 'Salih Özcan',
    'Samet Akaydin', 'Yusuf Yazıcı', 'İsmail Yüksek', 'Cenk Tosun', 'Yunus Akgün'
]
touches = ["Pass", "Ball Receipt*", "Carry", "Clearance", "Foul Won", "Block", "Ball Recovery", "Duel", "Dribbl

# Tüm oyuncuların verilerini filtreleme
player_dfs = {}
for player in players:
    player_df = events_all_df[(events_all_df.player == player) & (events_all_df.type.isin(touches))]
    player_dfs[player] = player_df

# Figür ve eksenleri oluşturma
num_players = len(players)
ncols = 4  # 4 sütun
nrows = (num_players + ncols - 1) // ncols  # Satır sayısını hesapla

fig, axs = plt.subplots(nrows=nrows, ncols=ncols, figsize=(25, 8*nrows), constrained_layout=True)  # Figür boyu
axs = axs.flatten()  # Eksenleri düzleştir

# Isı haritalarını oluşturma ve çizme
for i, (player, player_df) in enumerate(player_dfs.items()):
    ax = axs[i]
```

```
    bin_statistic = pitch.bin_statistic(player_df.x, player_df.y, statistic="count", bins=(6, 4), normalize=Tru

    # Maksimum ve minimum değerlerin belirlenmesi
    vmax = bin_statistic["statistic"].max()
    vmin = 0

    # Isı haritası oluşturma
    pitch.heatmap(bin_statistic, ax=ax, cmap=cmap, vmax=vmax, vmin=vmin)

    # Etiketleme ve başlık ekleme
    pitch.label_heatmap(bin_statistic, color="white", path_effects=path_eff, fontsize=25, ax=ax,
                        str_format="{:.0%}", ha="center", va="center", exclude_zeros=True)

    # Oyuncu ismini başlık olarak ekleme
    ax.set_title(f'{player}', fontsize=18, loc='center', pad=0.4)

    # Saha çizgilerini çizme
    pitch.draw(ax=ax)

# Boş alanları azaltmak için tüm eksenleri ayarladık
for ax in axs:
    ax.set_facecolor('white')  # Arka plan rengi

    # Çizgi rengini ve kalınlığını ayarla
    ax.spines['top'].set_color('black')
    ax.spines['top'].set_linewidth(6)
    ax.spines['bottom'].set_color('black')
    ax.spines['bottom'].set_linewidth(6)
    ax.spines['left'].set_color('black')
    ax.spines['left'].set_linewidth(6)
    ax.spines['right'].set_color('black')
    ax.spines['right'].set_linewidth(6)

    ax.set_xticks([])  # X eksenindeki işaretleri kaldır
    ax.set_yticks([])  # Y eksenindeki işaretleri kaldır

fig.suptitle('Oyuncu Faaliyetlerinin Bölgesel Dağılımı', fontsize=24, fontweight='bold', y=1.02)

plt.tight_layout()
plt.show()
```





In [111…

```
colour1 = "white"
colour2 = "#c3c3c3"
colour3 = "#e21017"
cmaplist = [colour1, colour2, colour3]
cmap = LinearSegmentedColormap.from_list("", cmaplist)

# Path effect ayarları
path_eff = [path_effects.Stroke(linewidth=2, foreground="black"), path_effects.Normal()]

# saha olusturma
pitch = VerticalPitch(pitch_type="statsbomb", line_zorder=2, line_color="#000000", linewidth=2, half=False)

# Türkiye kadrosunu filtreleme
players = [
    'Abdülkerim Bardakcı', 'Arda Güler', 'Barış Alper Yılmaz', 'Fehmi Mert Günok', 'Ferdi Erenay Kadıoğlu',
    'Hakan Çalhanoğlu', 'Kaan Ayhan', 'Kenan Yildiz', 'Mehmet Zeki Çelik', 'Merih Demiral',
    'Mert Müldür', 'Muhammed Kerem Aktürkoğlu', 'Okay Yokuşlu', 'Orkun Kökçü', 'Salih Özcan',
    'Samet Akaydin', 'Yusuf Yazıcı', 'İsmail Yüksek', 'Cenk Tosun', 'Yunus Akgün'
]
touches = ["Pass"]

# Tüm oyuncuların verilerini filtreleme
player_dfs = {}
for player in players:
    player_df = events_all_df[(events_all_df.player == player) & (events_all_df.type.isin(touches))]
    player_dfs[player] = player_df

# Figür ve eksenleri oluşturma
num_players = len(players)
ncols = 4  # 4 sütun
nrows = (num_players + ncols - 1) // ncols  # Satır sayısını hesapla

fig, axs = plt.subplots(nrows=nrows, ncols=ncols, figsize=(25, 8*nrows), constrained_layout=True)  # Figür boyu
axs = axs.flatten()  # Eksenleri düzleştir

# Isı haritalarını oluşturma ve çizme
for i, (player, player_df) in enumerate(player_dfs.items()):
    ax = axs[i]

    bin_statistic = pitch.bin_statistic(player_df.x, player_df.y, statistic="count", bins=(6, 4), normalize=Tru
```

```
    # Maksimum ve minimum değerlerin belirlenmesi
    vmax = bin_statistic["statistic"].max()
    vmin = 0

    # Isı haritası oluşturma
    pitch.heatmap(bin_statistic, ax=ax, cmap=cmap, vmax=vmax, vmin=vmin)

    # Etiketleme ve başlık ekleme
    pitch.label_heatmap(bin_statistic, color="white", path_effects=path_eff, fontsize=25, ax=ax,
                        str_format="{:.0%}", ha="center", va="center", exclude_zeros=True)

    # Oyuncu ismini başlık olarak ekleme
    ax.set_title(f'{player}', fontsize=18, loc='center', pad=0.4)

    # Saha çizgilerini çizme
    pitch.draw(ax=ax)

# Boş alanları azaltmak için tüm eksenleri ayarladık
for ax in axs:
    ax.set_facecolor('white')  # Arka plan rengi

    # Çizgi rengini ve kalınlığını ayarla
    ax.spines['top'].set_color('black')
    ax.spines['top'].set_linewidth(6)
    ax.spines['bottom'].set_color('black')
    ax.spines['bottom'].set_linewidth(6)
    ax.spines['left'].set_color('black')
    ax.spines['left'].set_linewidth(6)
    ax.spines['right'].set_color('black')
    ax.spines['right'].set_linewidth(6)

    ax.set_xticks([])  # X eksenindeki işaretleri kaldır
    ax.set_yticks([])  # Y eksenindeki işaretleri kaldır

fig.suptitle('Türkiye Futbol Takımı Pas Isı Haritaları', fontsize=24, fontweight='bold', y=1.02)

plt.tight_layout()
plt.show()
```

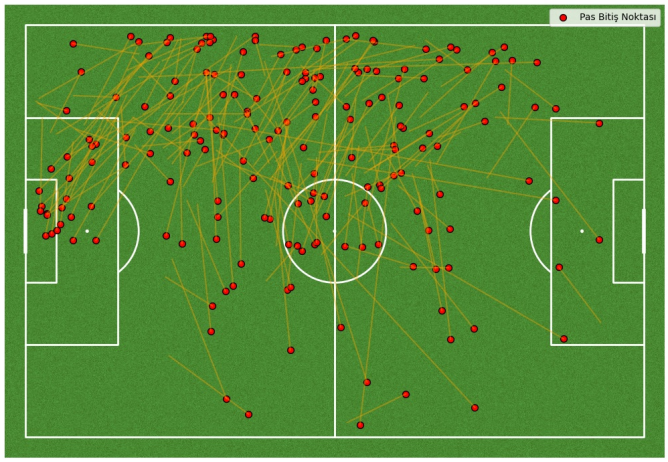
In [113…

```
# Türkiye kadrosu
players = [
    'Abdülkerim Bardakcı', 'Arda Güler', 'Barış Alper Yılmaz', 'Fehmi Mert Günok', 'Ferdi Erenay Kadıoğlu',
    'Hakan Çalhanoğlu', 'Kaan Ayhan', 'Kenan Yildiz',  'Merih Demiral','Mert Müldür', 'Samet Akaydin','Orkun Kö
    'Okay Yokuşlu', 'Salih Özcan','İsmail Yüksek', 'Mehmet Zeki Çelik','Yusuf Yazıcı', 'Cenk Tosun', 'Yunus Akg
]

# Figür ve eksenleri oluşturma
num_players = len(players)
ncols = 2  # 2 sütun
nrows = (num_players + ncols - 1) // ncols  # Satır sayısını hesapla

# Figür boyutunu artırarak daha fazla alan sağlama
fig, axs = plt.subplots(nrows=nrows, ncols=ncols, figsize=(20, 9*nrows), constrained_layout=True)
axs = axs.flatten()  # Eksenleri düzleştir

# Her oyuncu için isabetli pasları çizme
for i, player in enumerate(players):
    ax = axs[i]

    # Oyuncunun isabetli paslarını filtreleme
    player_passes = events_all_df[
        (events_all_df.player == player) &
        (events_all_df.type == "Pass") &
        (events_all_df.pass_outcome.isna())
    ]

    # Saha çizgilerini çizme
    pitch = Pitch(pitch_type='statsbomb', pitch_color='grass', line_color='white')
    pitch.draw(ax=ax)  # Saha boyutunu figür boyutuna uyacak şekilde ayarla

    # Pasların başlangıç ve bitiş noktalarını çizme
    for _, row in player_passes.iterrows():
        ax.plot(
            [row['x'], row['pass_end_x']],
            [row['y'], row['pass_end_y']],
            color='orange', linestyle='-', linewidth=1.5, alpha=0.4
        )

    # Pasların bitiş noktalarını işaretleme
    ax.scatter(
        x=player_passes.pass_end_x,
```

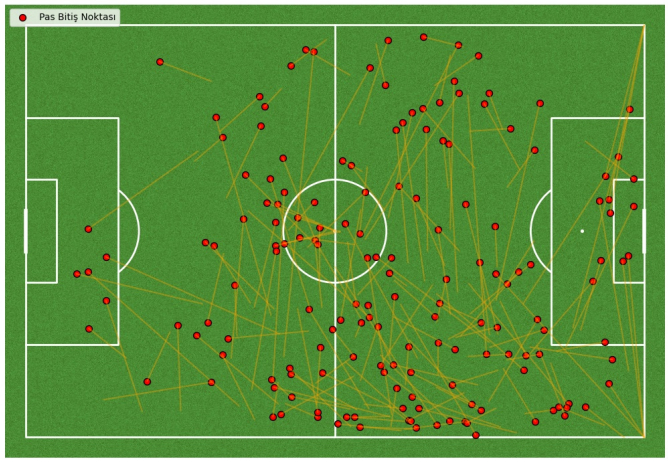
```
        y=player_passes.pass_end_y,
        color='red',
        s=50,
        edgecolor='black',
        marker='o',
        label='Pas Bitiş Noktası'
    )

    # Başlık ve etiket ayarları
    ax.set_title(f'{player}', fontsize=23)



    # İşaretlerin tekrar etmesini engelleme
    handles, labels = ax.get_legend_handles_labels()
    by_label = dict(zip(labels, handles))
    ax.legend(by_label.values(), by_label.keys())

fig.suptitle('Türkiye Futbol Takımı İsabetli Pas Haritaları', fontsize=24, fontweight='bold', y=1.02)

plt.show()
```

Türkiye Futbol Takımı İsabetli Pas Haritaları

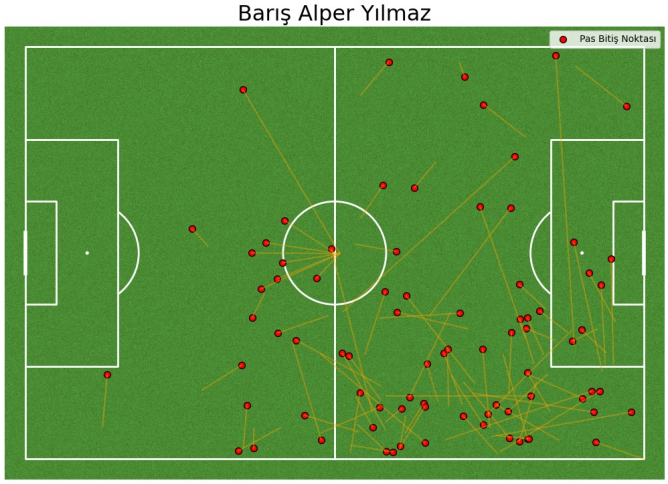
Abdülkerim Bardakcı



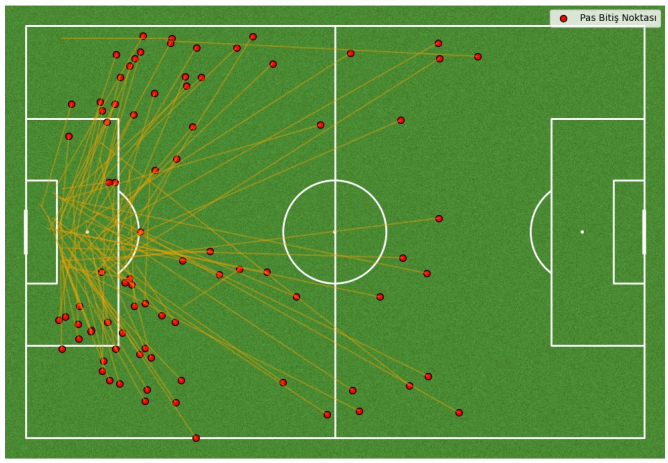
Arda Güler



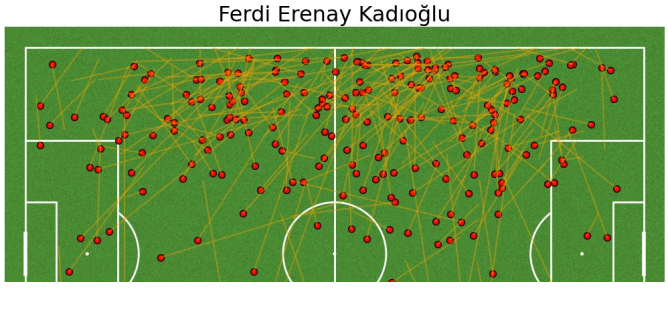
Barış Alper Yılmaz



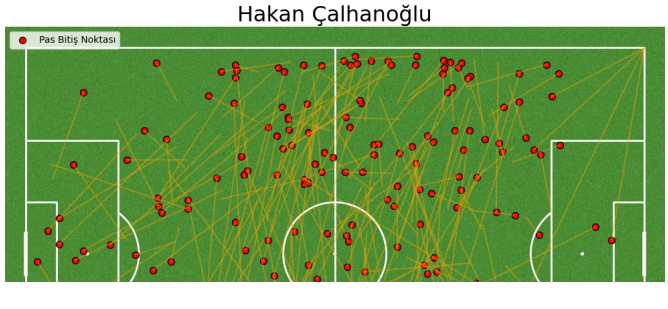
Fehmi Mert Günok

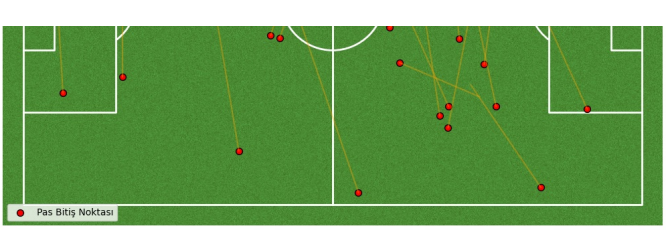


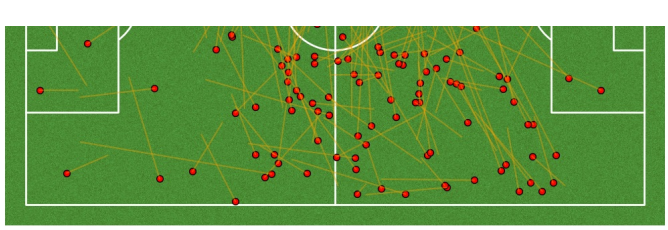
Ferdi Erenay Kadıoğlu



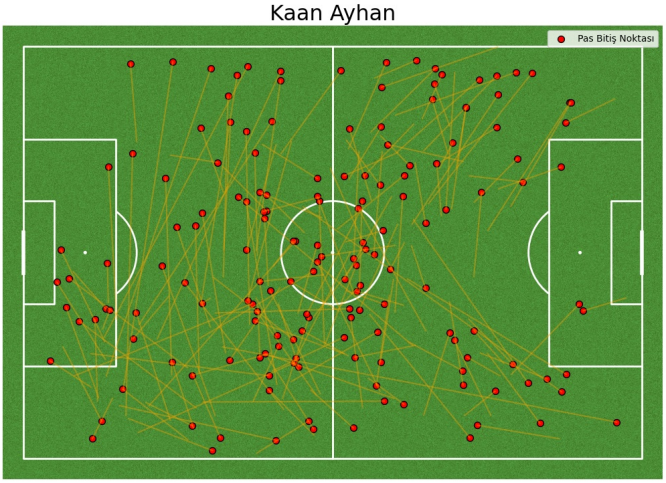
Hakan Çalhanoğlu



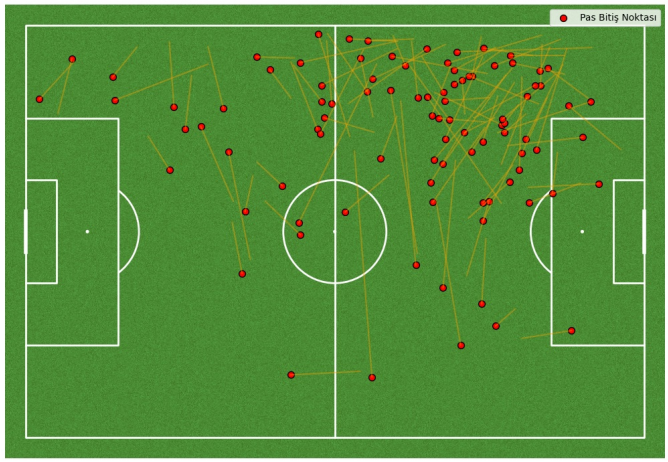




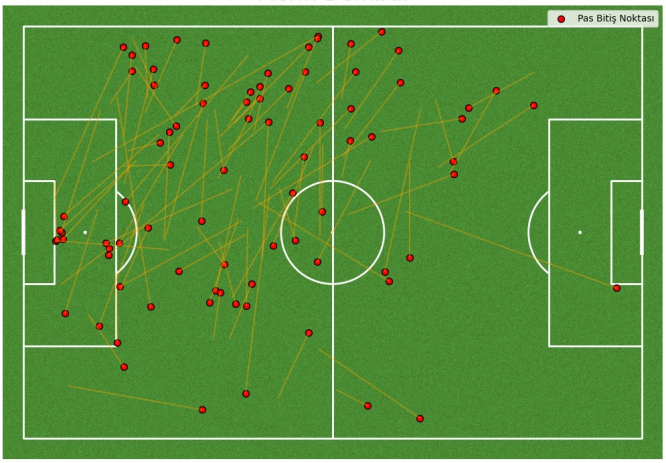
Kaan Ayhan



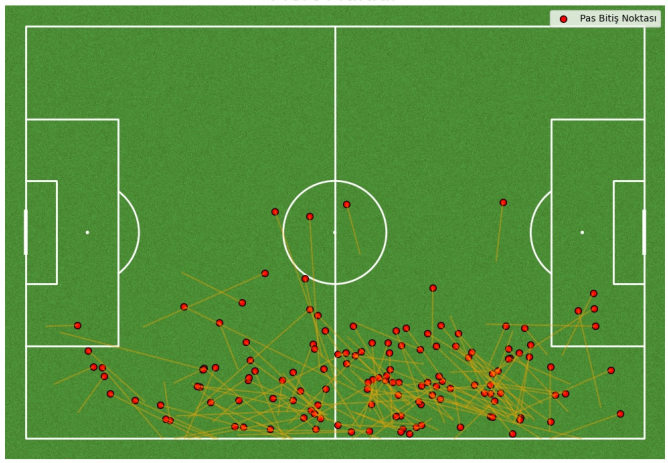
Kenan Yildiz



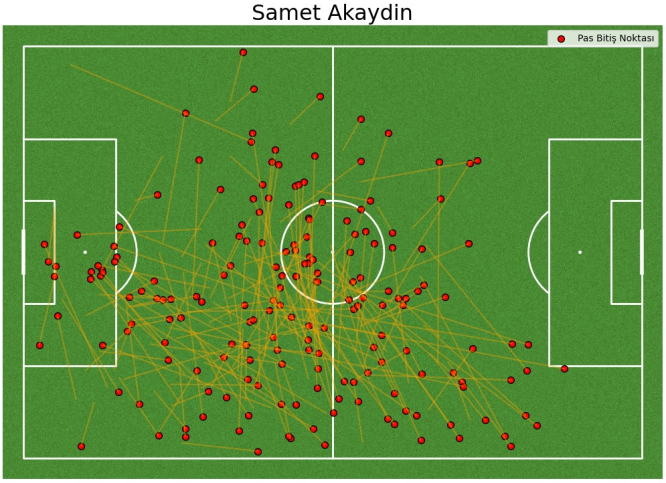
Merih Demiral



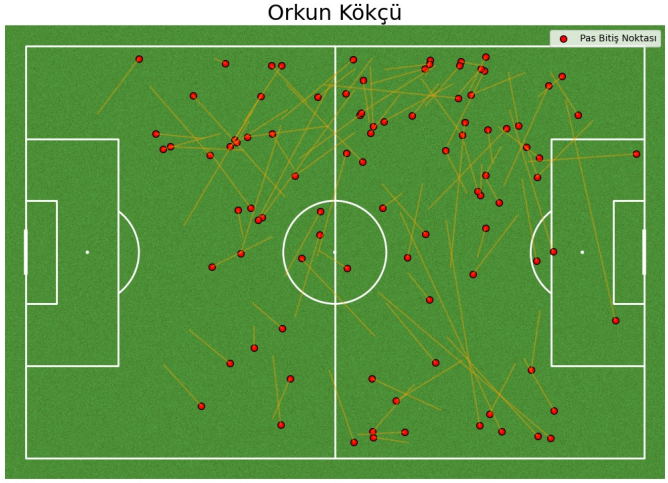
Mert Müldür



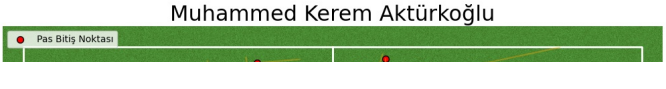
Samet Akaydin



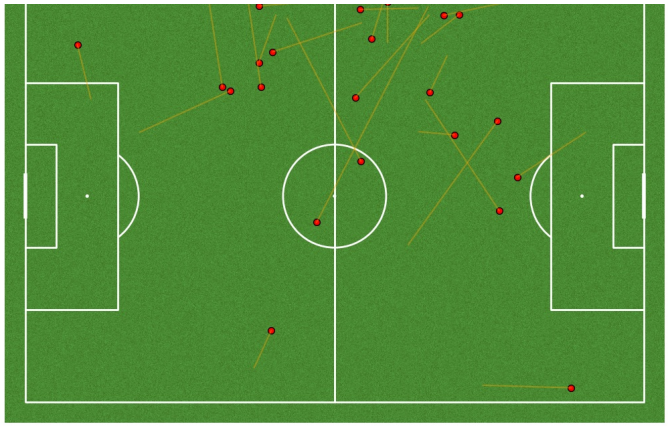
Orkun Kökçü

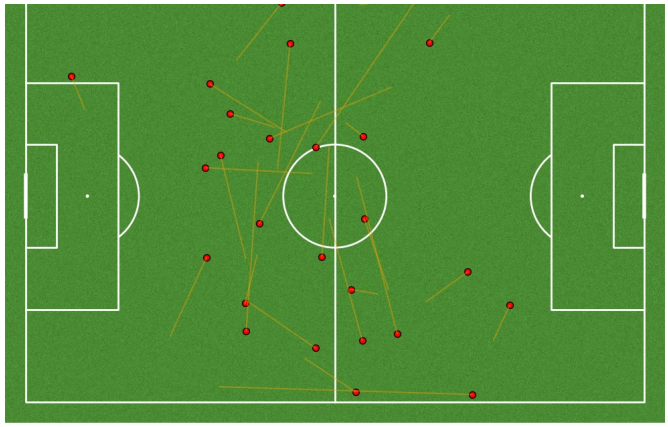


Muhammed Kerem Aktürkoğlu

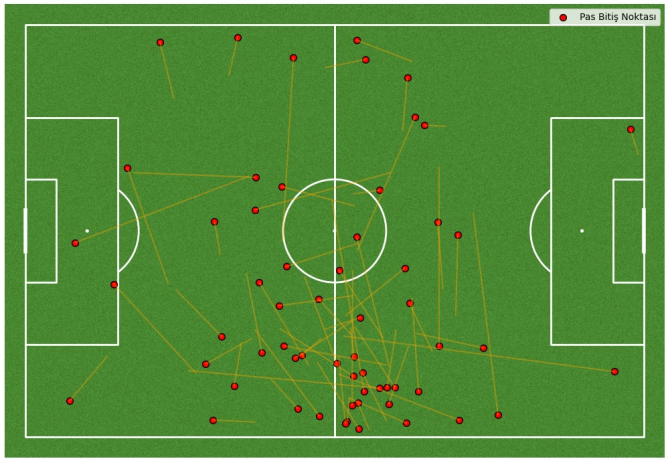


Okay Yokuşlu

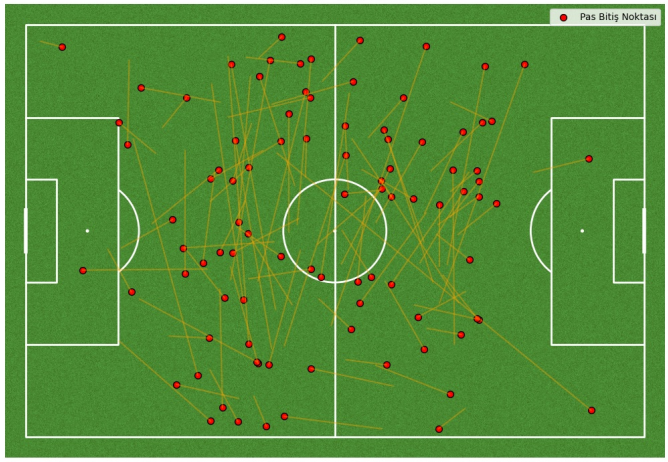




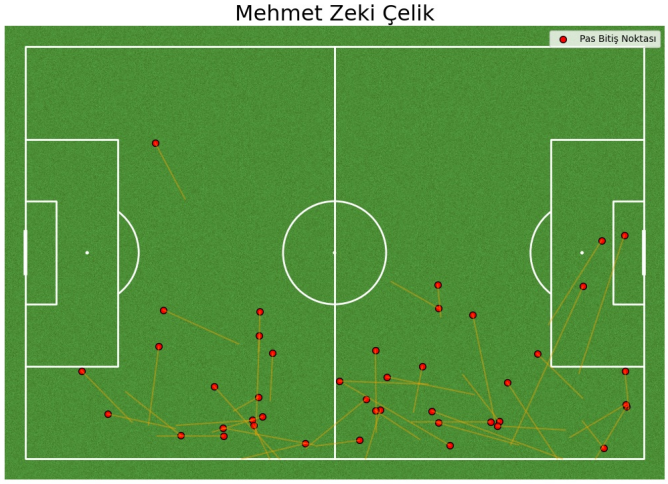
Salih Özcan



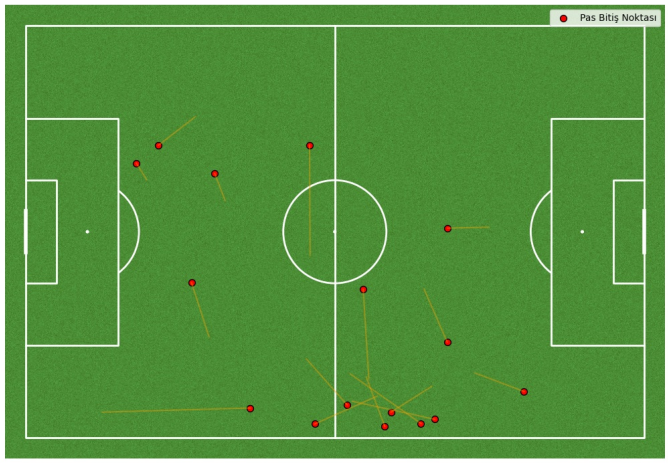
İsmail Yüksek



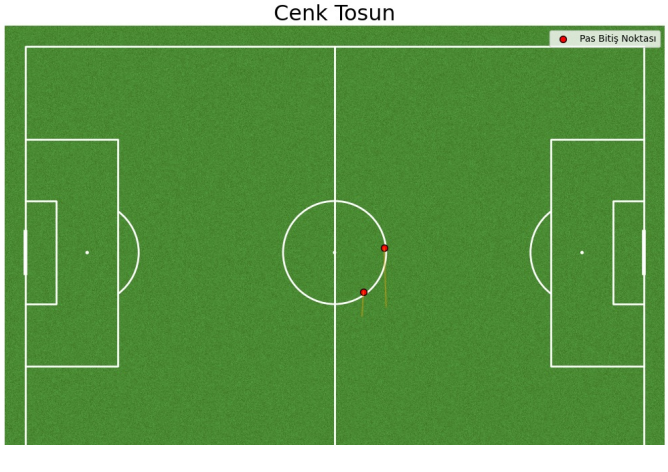
Mehmet Zeki Çelik



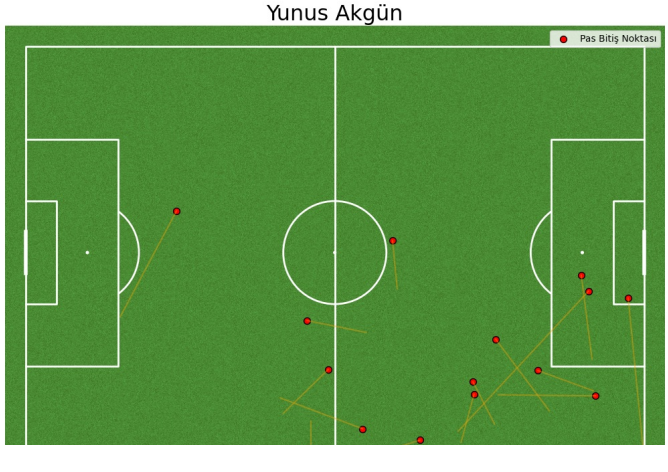
Yusuf Yazıcı

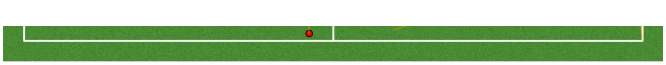


Cenk Tosun



Yunus Akgün



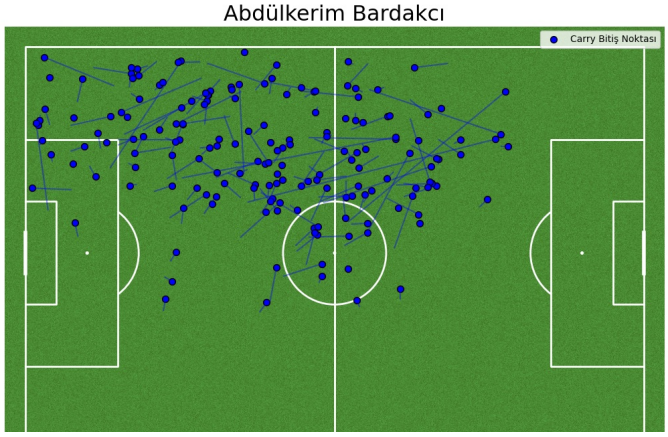



In [115…

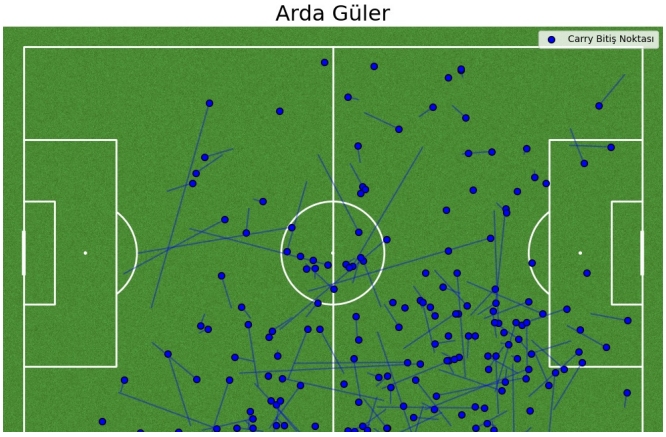
```
# Türkiye kadrosu
players = [
    'Abdülkerim Bardakcı', 'Arda Güler', 'Barış Alper Yılmaz', 'Fehmi Mert Günok', 'Ferdi Erenay Kadıoğlu',
    'Hakan Çalhanoğlu', 'Kaan Ayhan', 'Kenan Yildiz',  'Merih Demiral', 'Mert Müldür', 'Samet Akaydin',
    'Orkun Kökçü', 'Muhammed Kerem Aktürkoğlu', 'Okay Yokuşlu', 'Salih Özcan', 'İsmail Yüksek',
    'Mehmet Zeki Çelik', 'Yusuf Yazıcı', 'Cenk Tosun', 'Yunus Akgün'
]

# Figür ve eksenleri oluşturma
num_players = len(players)
ncols = 2  # 2 sütun
nrows = (num_players + ncols - 1) // ncols  # Satır sayısını hesapla

# Figür boyutunu artırarak daha fazla alan sağlama
fig, axs = plt.subplots(nrows=nrows, ncols=ncols, figsize=(20, 9*nrows), constrained_layout=True)
axs = axs.flatten()  # Eksenleri düzleştir

# Her oyuncu için carry hareketlerini çizme
for i, player in enumerate(players):
    ax = axs[i]

    # Oyuncunun carry hareketlerini filtreleme
    player_carries = events_all_df[
        (events_all_df.player == player) &
        (events_all_df['type'] == 'Carry')
    ]

    # Saha çizgilerini çizme
    pitch = Pitch(pitch_type='statsbomb', pitch_color='grass', line_color='white')
    pitch.draw(ax=ax)  # Saha boyutunu figür boyutuna uyacak şekilde ayarla

    # Carry hareketlerinin başlangıç ve bitiş noktalarını çizme
    for _, row in player_carries.iterrows():
        ax.plot(
            [row['x'], row['carry_end_x']],
            [row['y'], row['carry_end_y']],
            color='blue', linestyle='-', linewidth=1.5, alpha=0.4
        )

    # Carry hareketlerinin bitiş noktalarını işaretleme
    ax.scatter(
        x=player_carries.carry_end_x,
        y=player_carries.carry_end_y,
        color='blue',
        s=50,
        edgecolor='black',
        marker='o',
        label='Carry Bitiş Noktası'
    )

    # Başlık ve etiket ayarları
    ax.set_title(f'{player}', fontsize=23)

    # İşaretlerin tekrar etmesini engelleme
    handles, labels = ax.get_legend_handles_labels()
    by_label = dict(zip(labels, handles))
    ax.legend(by_label.values(), by_label.keys())

fig.suptitle('Türkiye Futbol Takımı Top Sürme Haritası', fontsize=24, fontweight='bold', y=1.02)

plt.show()
```

Türkiye Futbol Takımı Top Sürme Haritası

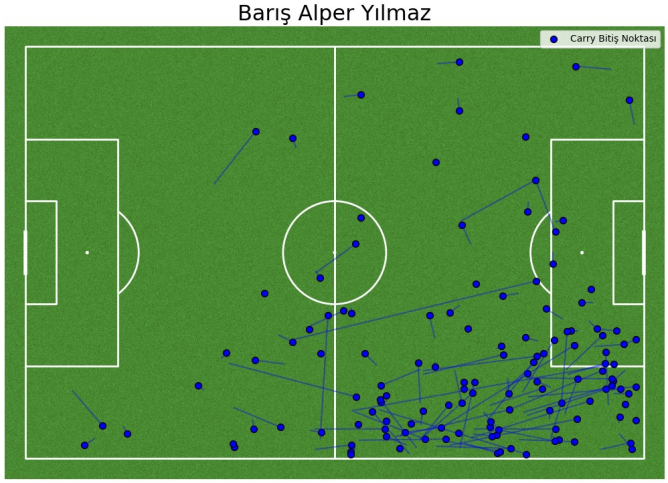
Abdülkerim Bardakcı



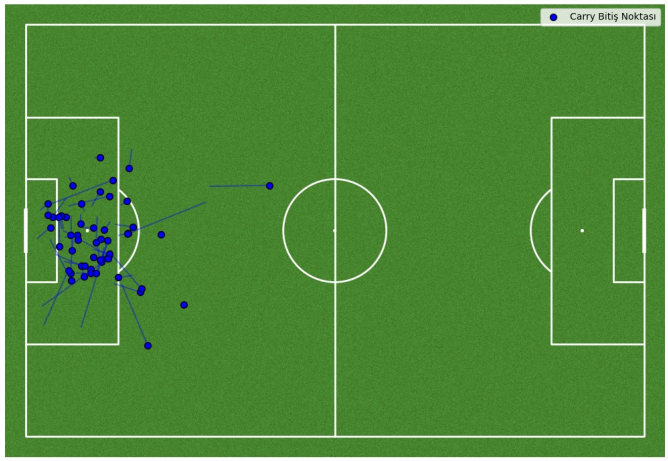
Arda Güler



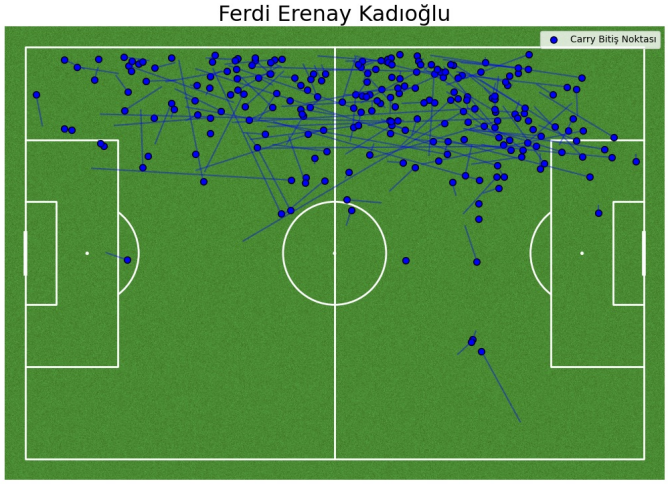
Barış Alper Yılmaz



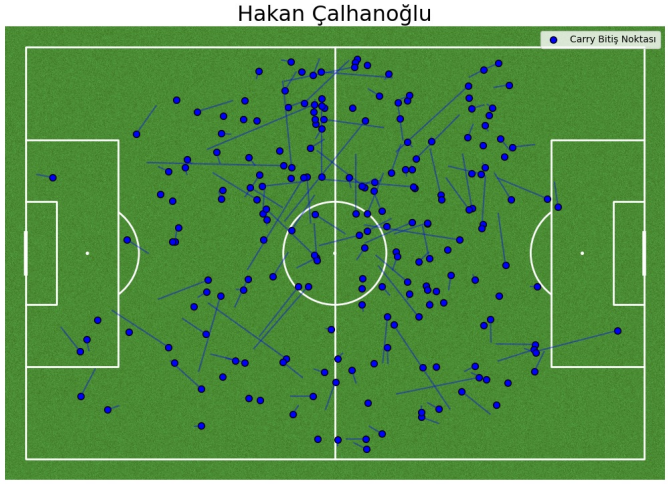
Fehmi Mert Günok



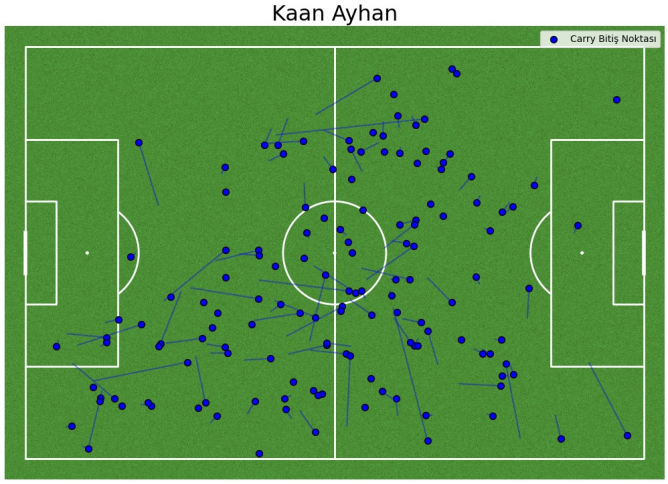
Ferdi Erenay Kadıoğlu



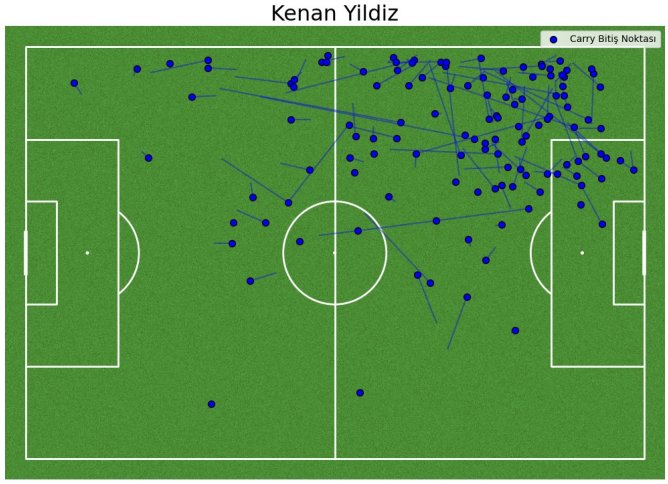
Hakan Çalhanoğlu



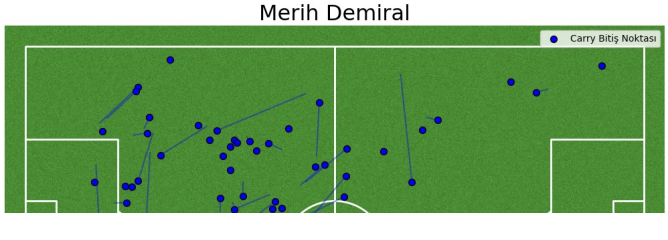
Kaan Ayhan



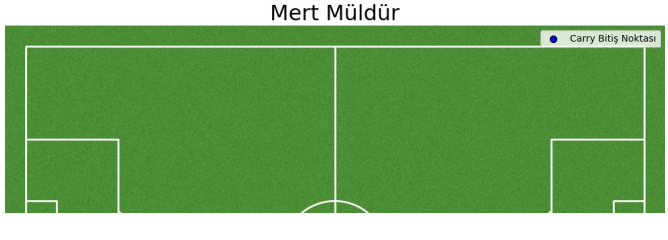
Kenan Yildiz

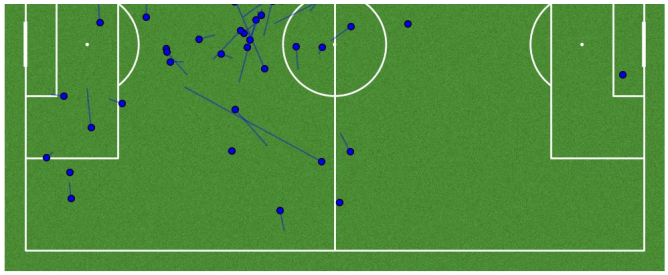


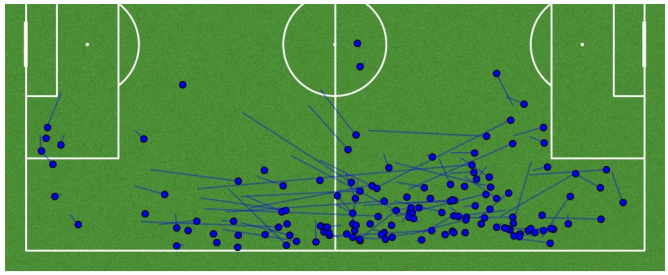
Merih Demiral



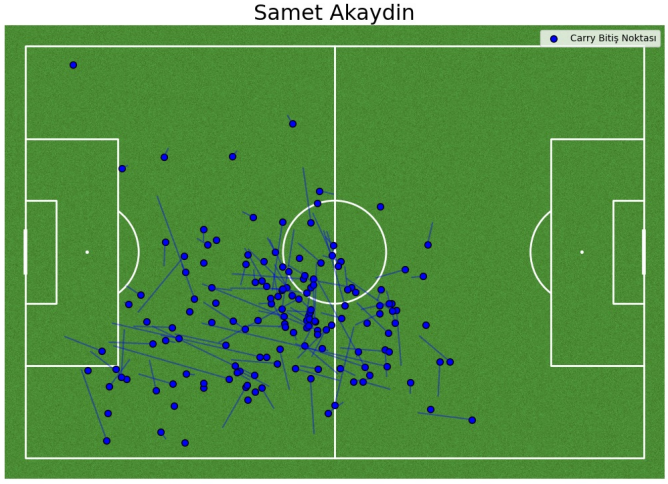
Mert Müldür



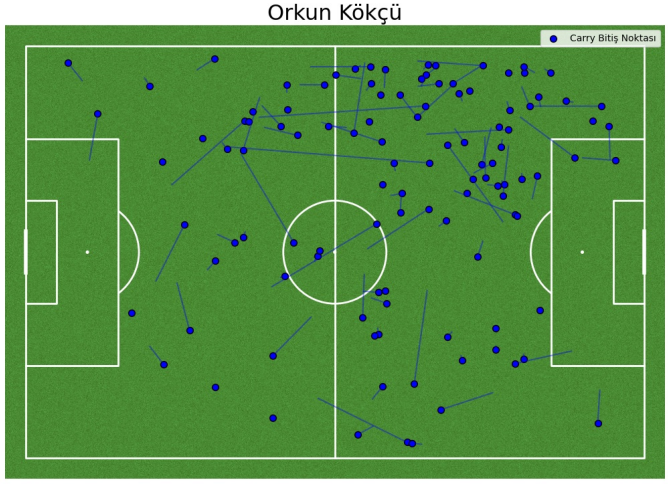




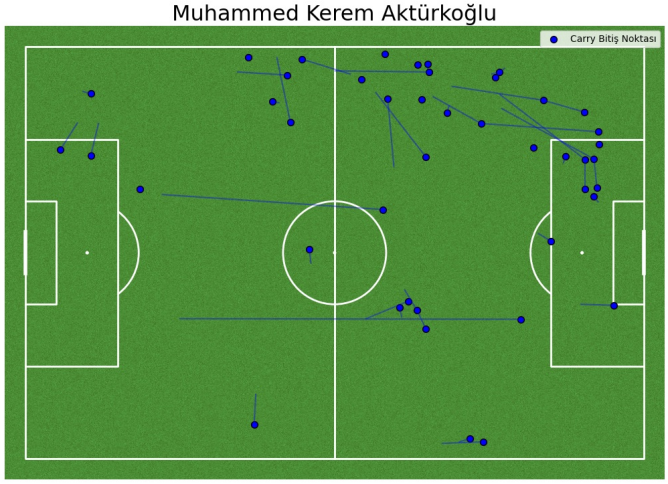
Samet Akaydin



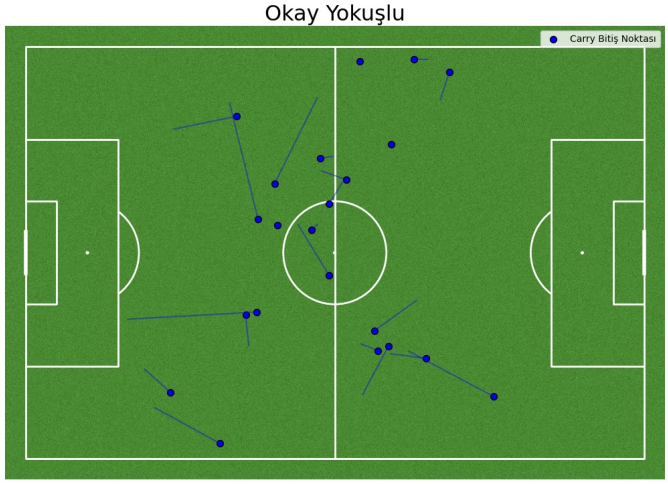
Orkun Kökçü



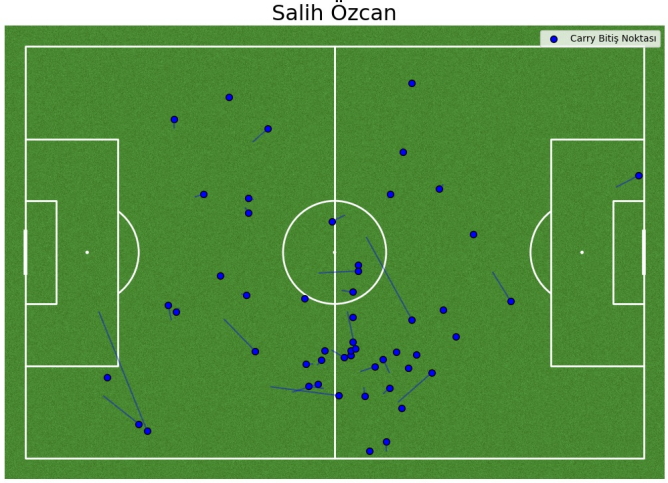
Muhammed Kerem Aktürkoğlu



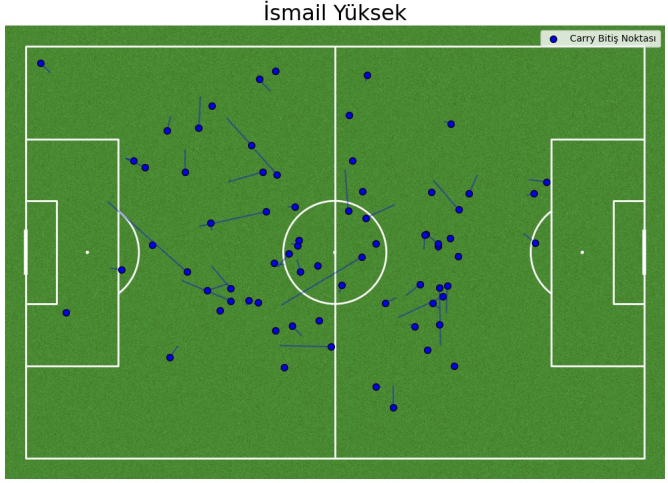
Okay Yokuşlu



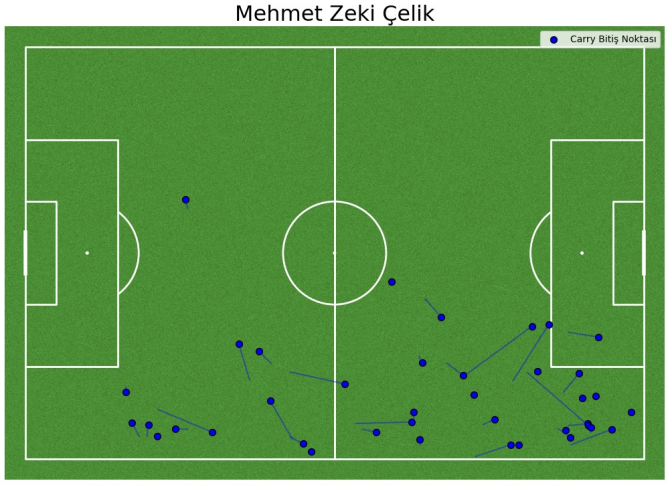
Salih Özcan



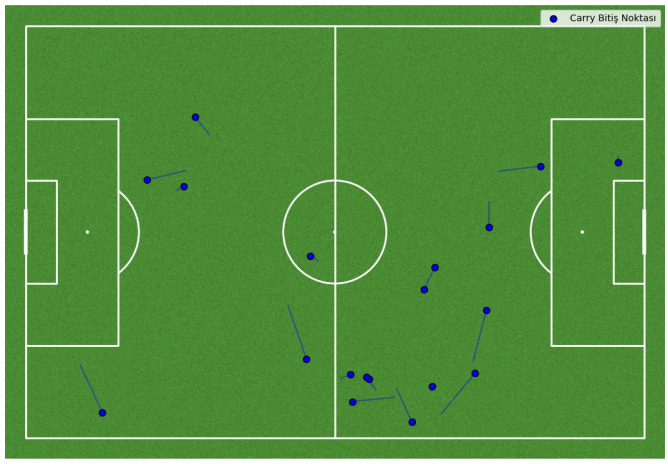
İsmail Yüksek



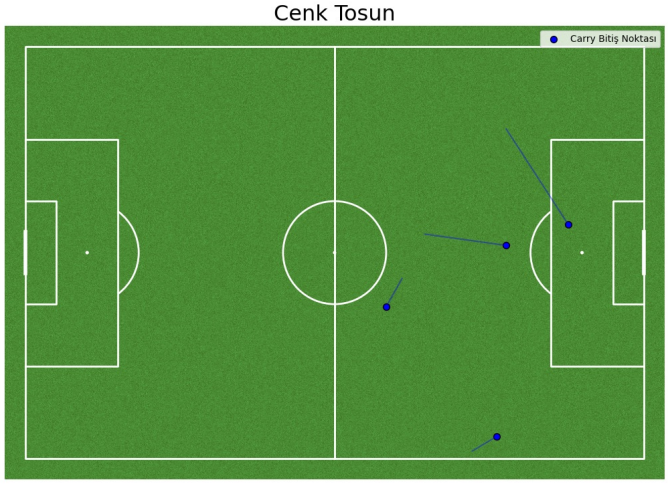
Mehmet Zeki Çelik



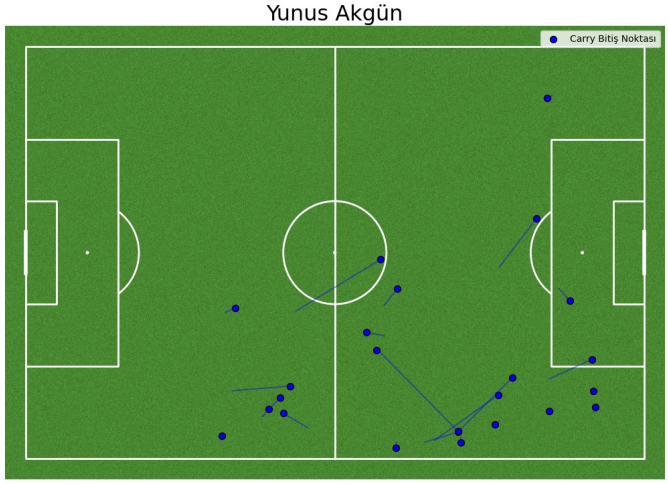
Yusuf Yazıcı



Cenk Tosun



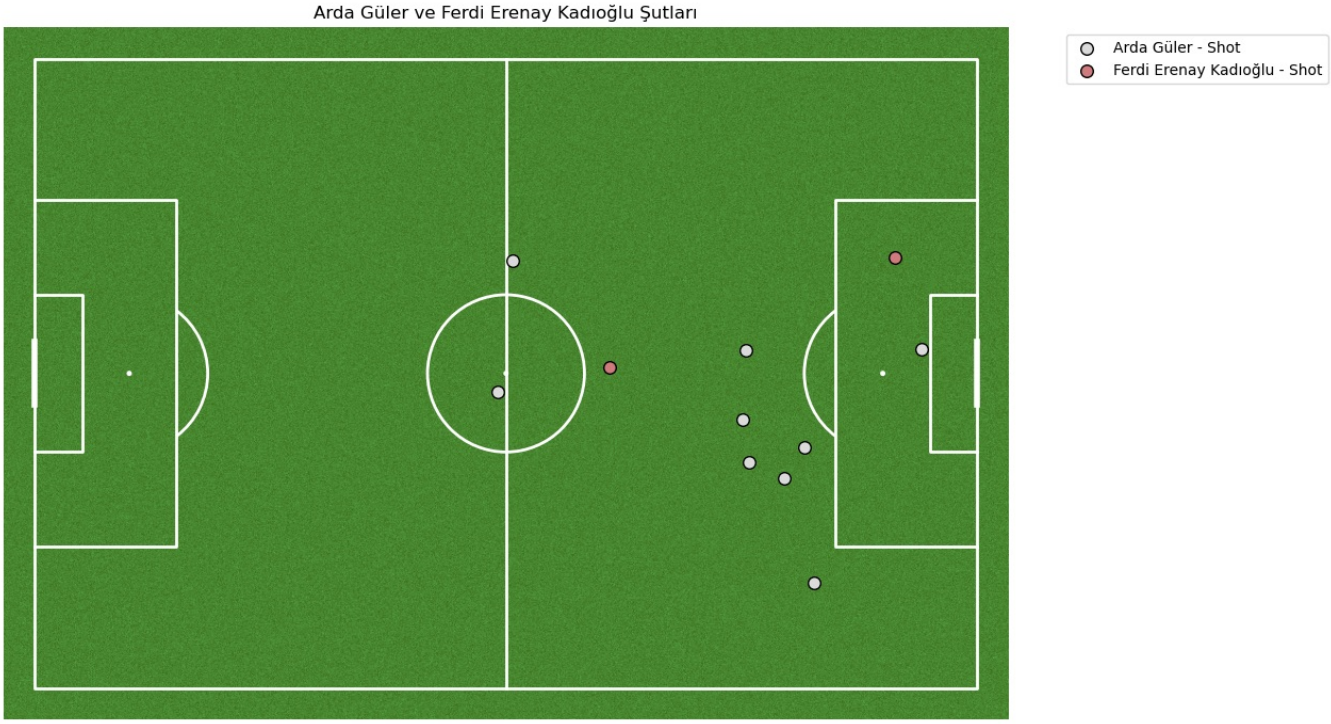
Yunus Akgün



In [121…

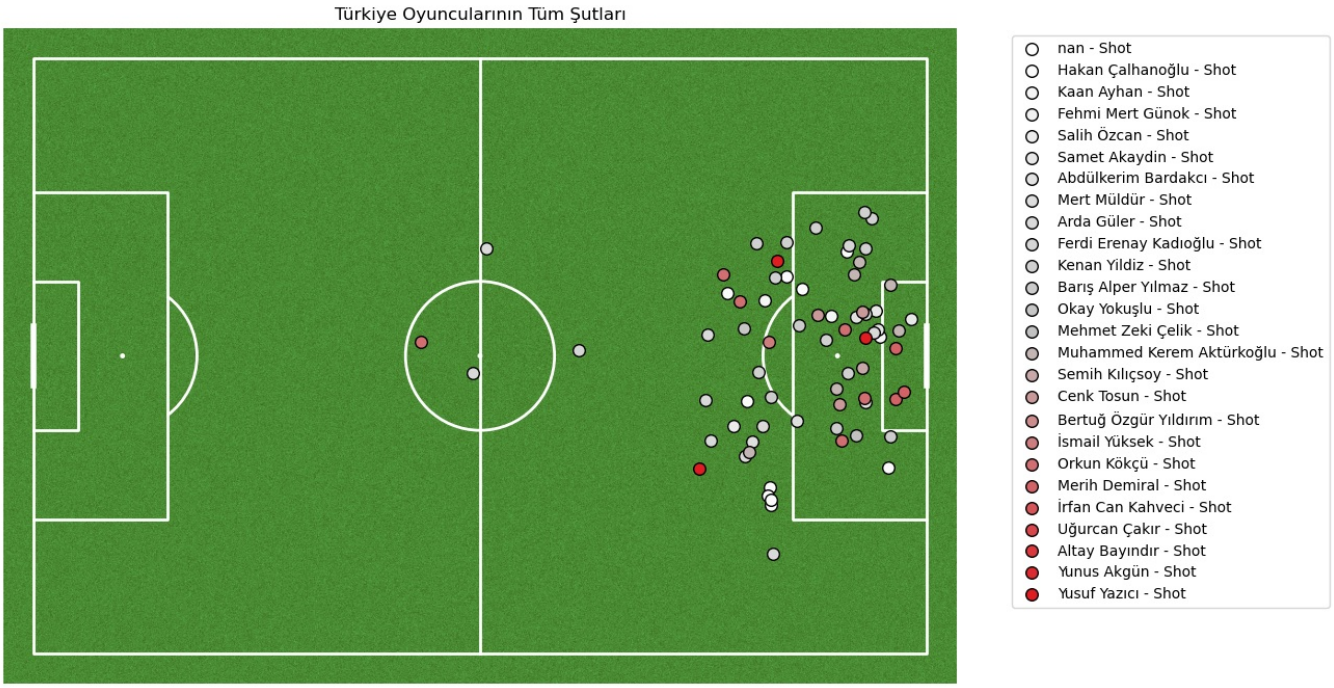
```python
# Oyuncular ve değişken
player1 = "Arda Güler"
player2 = "Ferdi Erenay Kadıoğlu"
touches = ["Shot"]

# Veri çerçevelerini filtreleme
player1_df = events_all_df[(events_all_df['player'] == player1) & (events_all_df['type'].isin(touches))]
player2_df = events_all_df[(events_all_df['player'] == player2) & (events_all_df['type'].isin(touches))]

# Renkler ve renk haritası
colour1 = "white"
colour2 = "#c3c3c3"
colour3 = "#e21017"
cmaplist = [colour1, colour2, colour3]
cmap = LinearSegmentedColormap.from_list("", cmaplist)

# Pitch objesi oluşturma
pitch = Pitch(pitch_type='statsbomb', pitch_color='grass', line_color='white')
fig, ax = pitch.draw(figsize=(15, 7))

# Ferdi
for event_type in touches:
    event_df = player1_df[player1_df['type'] == event_type]
    ax.scatter(
        x=event_df['x'],
        y=event_df['y'],
        color=cmap(0.3),
        s=70,
        edgecolor='black',
        label=f'{player1} - {event_type}'
    )

# Arda
for event_type in touches:
    event_df = player2_df[player2_df['type'] == event_type]
    ax.scatter(
        x=event_df['x'],
        y=event_df['y'],
        color=cmap(0.7),
        s=70,
        edgecolor='black',
        label=f'{player2} - {event_type}'
    )
```

```
# İşaretlerin tekrar etmesini engelleme
handles, labels = ax.get_legend_handles_labels()
by_label = dict(zip(labels, handles))
ax.legend(by_label.values(), by_label.keys(), bbox_to_anchor=(1.05, 1), loc='upper left')

plt.title(f'{player1} ve {player2} Şutları')
plt.show()
```



In [123…

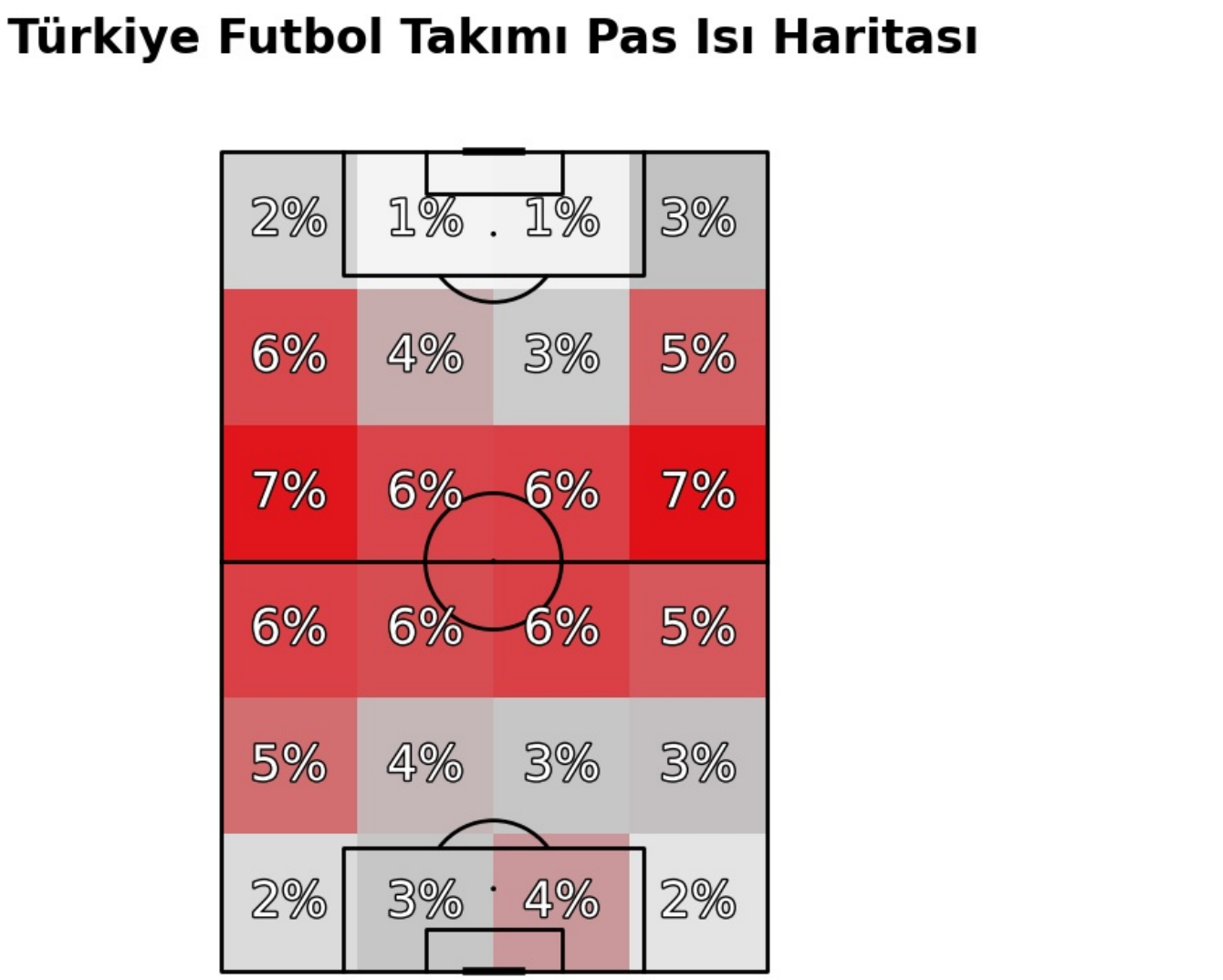
```
turkish_players = events_all_df[events_all_df['team'] == 'Turkey']['player'].unique()

# Değişken
touches = ["Shot"]

# Renkler ve renk haritası
colour1 = "white"
colour2 = "#c3c3c3"
colour3 = "#e21017"
cmaplist = [colour1, colour2, colour3]

# Saha oluşturma
pitch = Pitch(pitch_type='statsbomb', pitch_color='grass', line_color='white')
fig, ax = pitch.draw(figsize=(15, 7))

for player in turkish_players:
    player_df = events_all_df[(events_all_df['player'] == player) & (events_all_df['type'].isin(touches))]

    for event_type in touches:
        event_df = player_df[player_df['type'] == event_type]

        # Renk tonu seçimi
        color_index = turkish_players.tolist().index(player) / len(turkish_players)
        ax.scatter(
            x=event_df['x'],
            y=event_df['y'],
            color=cmap(color_index),  # Oyuncuya göre renk tonu
            s=70,
            edgecolor='black',
            label=f'{player} - {event_type}'
        )

# İşaretlerin tekrar etmesini engelleme
handles, labels = ax.get_legend_handles_labels()
by_label = dict(zip(labels, handles))
ax.legend(by_label.values(), by_label.keys(), bbox_to_anchor=(1.05, 1), loc='upper left')

plt.title('Türkiye Oyuncularının Tüm Şutları')
plt.show()
```



In [125…

```python
import matplotlib.colors as mcolors

colour1 = "white"
colour2 = "#c3c3c3"
colour3 = "#e21017"
cmaplist = [colour1, colour2, colour3]
cmap = mcolors.LinearSegmentedColormap.from_list("", cmaplist)

# Path effect ayarları
path_eff = [path_effects.Stroke(linewidth=2, foreground="black"), path_effects.Normal()]

# Pitch objesi oluşturma
pitch = VerticalPitch(pitch_type="statsbomb", line_zorder=2, line_color="#000000", linewidth=2, half=False)

# Türkiye kadrosunu filtreleme
players = [
    'Abdülkerim Bardakcı', 'Arda Güler', 'Barış Alper Yılmaz', 'Fehmi Mert Günok', 'Ferdi Erenay Kadıoğlu',
    'Hakan Çalhanoğlu', 'Kaan Ayhan', 'Kenan Yildiz', 'Mehmet Zeki Çelik', 'Merih Demiral',
    'Mert Müldür', 'Muhammed Kerem Aktürkoğlu', 'Okay Yokuşlu', 'Orkun Kökçü', 'Salih Özcan',
    'Samet Akaydin', 'Yusuf Yazıcı', 'İsmail Yüksek', 'Cenk Tosun', 'Yunus Akgün'
]
touches = ["Pass"]

# Tüm oyuncuların verilerini filtreleme ve birleştirme
all_passes_df = pd.concat([
    events_all_df[(events_all_df['player'] == player) & (events_all_df['type'].isin(touches))]
    for player in players
])

# NaN değerlerini filtreleme
all_passes_df = all_passes_df.dropna(subset=['x', 'pass_end_x', 'pass_end_y'])  # 'x', 'pass_end_x', 'pass_end_

# Bin istatistiklerini hesaplama
bin_statistic = pitch.bin_statistic(all_passes_df['x'], all_passes_df['y'], statistic="count", bins=(6, 4), nor

# Maksimum ve minimum değerlerin belirlenmesi
vmax = bin_statistic["statistic"].max()
vmin = 0

# Figür ve eksenleri oluşturma
fig, ax = plt.subplots(figsize=(15, 7))

# Isı haritası oluşturma
pitch.heatmap(bin_statistic, ax=ax, cmap=cmap, vmax=vmax, vmin=vmin)

# Etiketleme ve başlık ekleme
pitch.label_heatmap(bin_statistic, color="white", path_effects=path_eff, fontsize=25, ax=ax,
                    str_format="{:.0%}", ha="center", va="center", exclude_zeros=True)

# Saha çizgilerini çizme
pitch.draw(ax=ax)

# Görselleştirme ayarları
ax.set_facecolor('white')  # Arka plan rengini beyaz yap

# Çizgi rengini ve kalınlığını ayarla
ax.spines['top'].set_color('black')
ax.spines['top'].set_linewidth(6)
ax.spines['bottom'].set_color('black')
```

```
ax.spines['bottom'].set_linewidth(6)
ax.spines['left'].set_color('black')
ax.spines['left'].set_linewidth(6)
ax.spines['right'].set_color('black')
ax.spines['right'].set_linewidth(6)

ax.set_xticks([])  # X eksenindeki işaretleri kaldır
ax.set_yticks([])  # Y eksenindeki işaretleri kaldır

fig.suptitle('Türkiye Futbol Takımı Pas Isı Haritası', fontsize=24, fontweight='bold', y=1.02)

plt.tight_layout()
plt.show()
```



In [127…
```
import pandas as pd

pass_counts = []

players = events_all_df['player'].unique()

for player in players:
    player_passes = events_all_df[
        (events_all_df.player == player) &
        (events_all_df.type == "Pass") &
        (events_all_df.pass_outcome.isna())
    ]
    num_passes = len(player_passes)
    pass_counts.append({'player': player, 'pass_count': num_passes})

pass_counts_df = pd.DataFrame(pass_counts)

pass_counts_df = pass_counts_df.sort_values(by='pass_count', ascending=False)
```

In [128…
```
pass_counts_df.head(10)
```

Out[128]:

	player	pass_count
19	Ferdi Erenay Kadıoğlu	240
4	Hakan Çalhanoğlu	229
13	Samet Akaydin	186
14	Abdülkerim Bardakcı	186
17	Arda Güler	163
5	Kaan Ayhan	163
16	Mert Müldür	137
39	Orkun Kökçü	94
20	Kenan Yildiz	93
38	İsmail Yüksek	92

In [131…
```
events_all_df["shot_outcome"].value_counts()
```

Out[131]:
```
shot_outcome
Off T            43
Blocked          42
Saved            29
Goal             14
Wayward          11
Post              2
Saved to Post     1
Name: count, dtype: int64
```

In [133…
```
player_stats = []

players = events_all_df['player'].unique()

for player in players:
    player_passes = events_all_df[
        (events_all_df.player == player) &
        (events_all_df.type == "Pass") &
        (events_all_df.pass_outcome.isna())
    ]
    num_passes = len(player_passes)

    player_carries = events_all_df[
        (events_all_df.player == player) &
        (events_all_df.type == "Carry")
    ]
    num_carries = len(player_carries)

    player_shots = events_all_df[
        (events_all_df.player == player) &
        (events_all_df.type == "Shot")
    ]
    num_shots = len(player_shots)

    player_stats.append({'player': player, 'pass_count': num_passes, 'carry_count': num_carries, 'shot_count':

player_stats_df = pd.DataFrame(player_stats)

player_stats_df['total'] = player_stats_df['pass_count'] + player_stats_df['carry_count'] + player_stats_df['sh

player_stats_df = player_stats_df.sort_values(by='total', ascending=False)
```

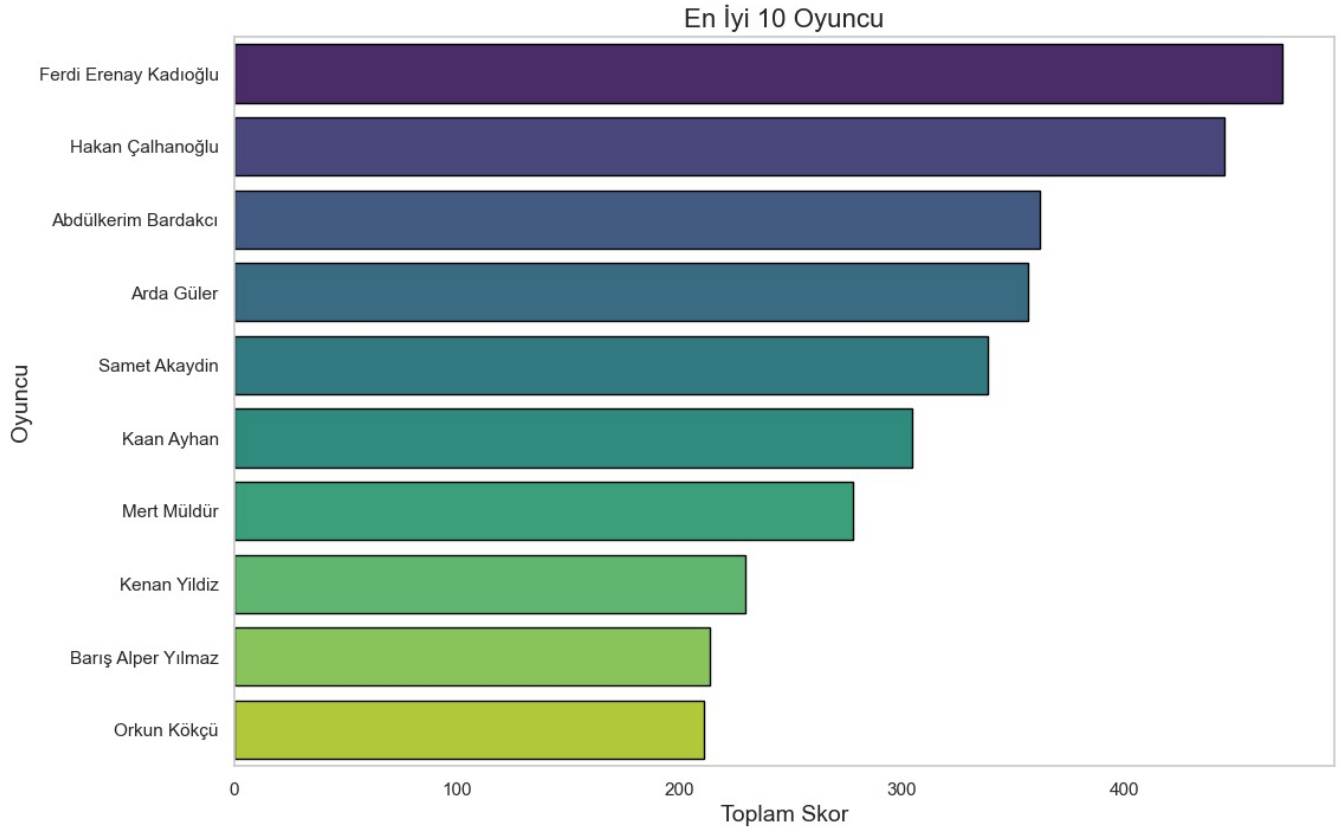
In [135…
```
player_stats_df.head(10)
```

Out[135]:

	player	pass_count	carry_count	shot_count	total
19	Ferdi Erenay Kadıoğlu	240	229	2	471
4	Hakan Çalhanoğlu	229	206	10	445
14	Abdülkerim Bardakcı	186	174	2	362
17	Arda Güler	163	185	9	357
13	Samet Akaydin	186	151	2	339
5	Kaan Ayhan	163	139	3	305
16	Mert Müldür	137	138	3	278
20	Kenan Yildiz	93	126	11	230
22	Barış Alper Yılmaz	81	129	4	214
39	Orkun Kökçü	94	111	6	211

In [137…
```
top_10_players = player_stats_df.sort_values(by='total', ascending=False).head(10)

plt.figure(figsize=(12, 8))
```

```
sns.set(style="whitegrid")

ax = sns.barplot(
    x='total',
    y='player',
    data=top_10_players,
    palette='viridis',
    edgecolor='black'
)

plt.title('En İyi 10 Oyuncu', fontsize=16)
plt.xlabel('Toplam Skor', fontsize=14)
plt.ylabel('Oyuncu', fontsize=14)
plt.grid(False)
plt.show()
```



In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

Bu kısım Uraz Akgül'den alınmıştır.

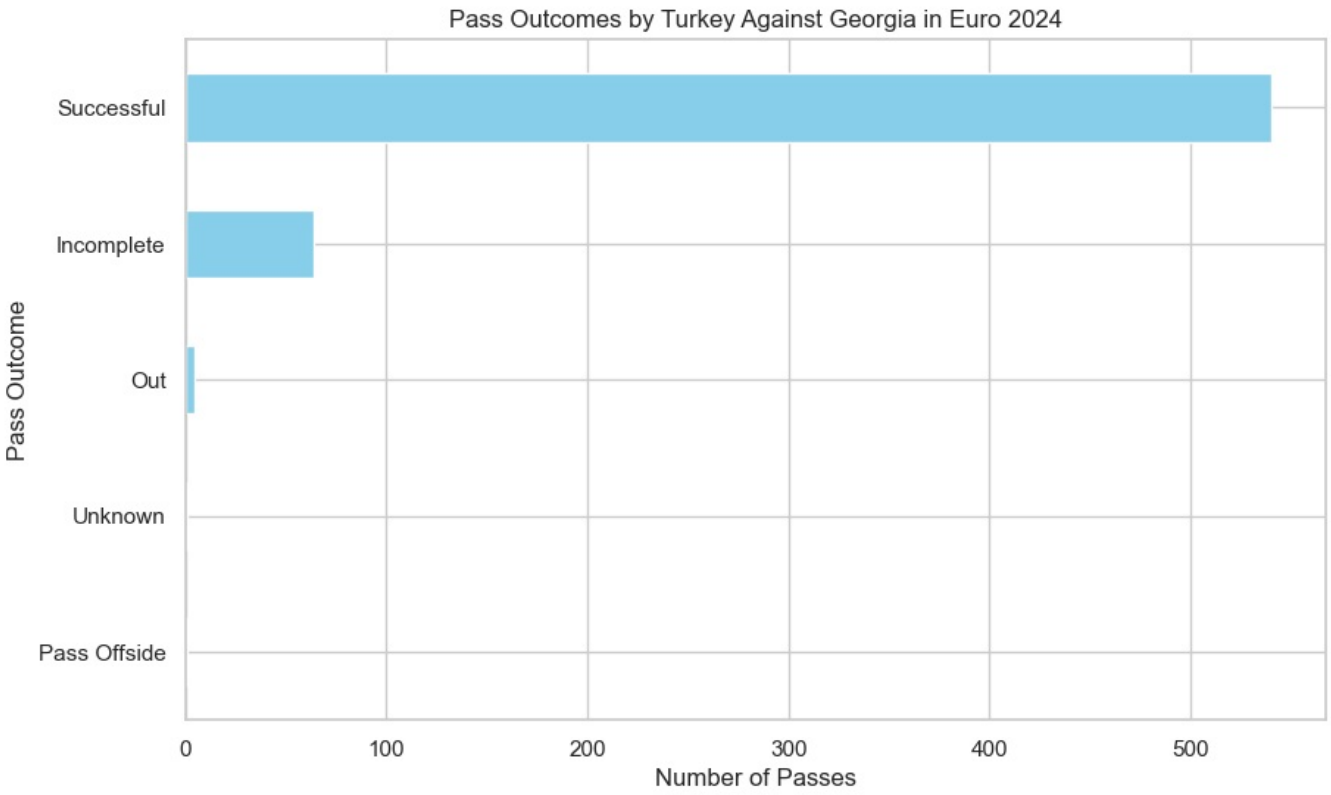
https://urazakgul.github.io/python-blog/posts.html

In [139…

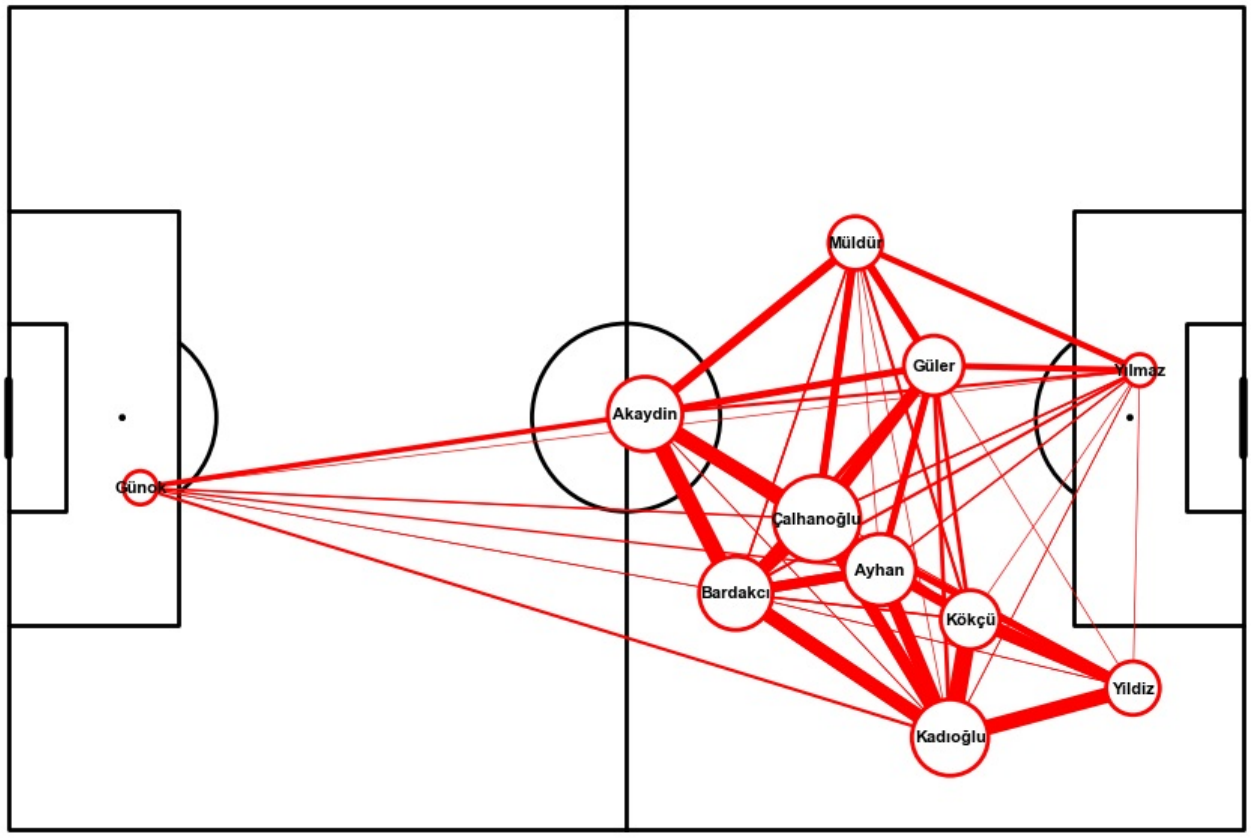
```
from mplsoccer import Pitch, Sbopen

# Türkiye için takım adı
team = 'Turkey'

# Maçlar ve ID'leri
matches = {
    'Georgia': 3938639,
    'Portugal': 3930174,
    'Czechia': 3930184,
    'Austria': 3941022,
    'Netherlands': 3942382
}
```

```python
for opponent, match_id in matches.items():
    # StatsBomb verilerini yükleme
    parser = Sbopen()
    df, related, freeze, tactics = parser.event(match_id)

    # Türkiye takımının olaylarını filtreleme
    df = df[df['team_name'] == team]

    # Pas olaylarını seçme ve NaN değerleri 'Successful' ile doldurma
    passes = df[df['type_name'] == 'Pass'].copy()
    passes['outcome_name'] = passes['outcome_name'].fillna('Successful')

    # Pas sonuçlarını görselleştirme
    plt.figure(figsize=(10, 6))
    pass_outcome_counts = passes['outcome_name'].value_counts().sort_values()
    pass_outcome_counts.plot(kind='barh', color='skyblue')
    plt.xlabel('Number of Passes')
    plt.ylabel('Pass Outcome')
    plt.title(f'Pass Outcomes by {team} Against {opponent} in Euro 2024')
    plt.show()

    # Başarılı pasları seçme
    successful_passes = passes[passes['outcome_name'] == 'Successful']

    # İlk oyuncu değişikliğinin dakikasını bulma
    first_sub = df[df['type_name'] == 'Substitution']['minute'].min()
    successful_passes = successful_passes[successful_passes['minute'] < first_sub]

    # Pas yapan oyuncuların ortalama konumlarını hesaplama
    passer_avg_loc = successful_passes.groupby('player_name').agg(
        avg_x=('x', 'mean'),
        avg_y=('y', 'mean'),
        pass_count=('id', 'size')
    ).reset_index()

    # Oyuncular arası pas akışını hesaplama
    pass_between = successful_passes.groupby(['player_name', 'pass_recipient_name']).size().reset_index(name='p

    # Ortalamaları ve akışları birleştirme
    pass_between = pass_between.merge(passer_avg_loc, left_on='player_name', right_on='player_name')
    pass_between = pass_between.merge(passer_avg_loc, left_on='pass_recipient_name', right_on='player_name', su

    # Oyuncu isimlerini sadeleştirme
    passer_avg_loc['player_name'] = passer_avg_loc['player_name'].apply(lambda name: name.split()[-1])
    pass_between['player_name'] = pass_between['player_name'].apply(lambda name: name.split()[-1])
    pass_between['pass_recipient_name'] = pass_between['pass_recipient_name'].apply(lambda name: name.split()[-

    # Futbol sahası görselleştirme
    pitch_length_x = 120
    pitch_width_y = 80
    pitch = Pitch(pitch_type='custom', pitch_length=pitch_length_x, pitch_width=pitch_width_y, line_color='blac
    fig, ax = pitch.draw(figsize=(10, 7))

    # Pas çizgilerini çizme
    pitch.lines(
        1.2 * pass_between.avg_x,
        0.8 * pass_between.avg_y,
        1.2 * pass_between.avg_x_end,
        0.8 * pass_between.avg_y_end,
        lw=pass_between.pass_between_count * 0.5,
        color='red',
        zorder=1,
        ax=ax
    )

    # Oyuncuların konumlarını ve pas sayılarını görselleştirme
    pitch.scatter(
        1.2 * passer_avg_loc.avg_x,
        0.8 * passer_avg_loc.avg_y,
        s=20 * passer_avg_loc['pass_count'].values,
        color='white',
        edgecolors='red',
        linewidth=2,
        alpha=1,
        zorder=1,
        ax=ax
    )

    # Oyuncu isimlerini ekleme
    for index, row in passer_avg_loc.iterrows():
        pitch.annotate(
            row['player_name'],
            xy=(1.2 * row.avg_x, 0.8 * row.avg_y),
            c='black',
            fontweight='bold',
            va='center',
            ha='center',
            size=8,
```

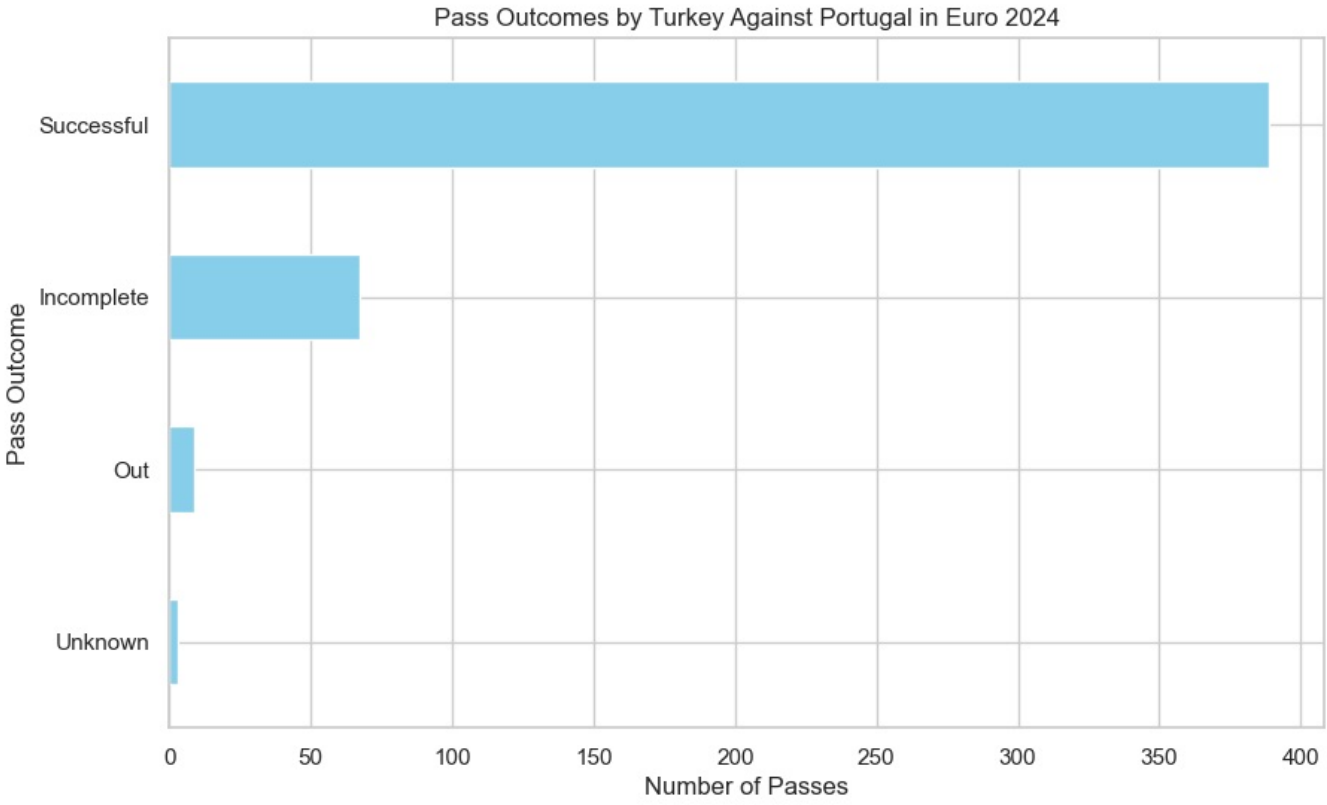
```
            ax=ax
        )

    # Başlık ve açıklama ekleme
    fig.suptitle(f"{team}'s Euro 2024 Pass Networks Against {opponent}", fontsize=16)
    ax.text(
        0.95, -0.05,
        f"First substitution occurred at {first_sub}'",
        color='gray',
        va='bottom',
        ha='right',
        fontsize=10,
        fontstyle='italic',
        transform=ax.transAxes
    )

    plt.show()
```



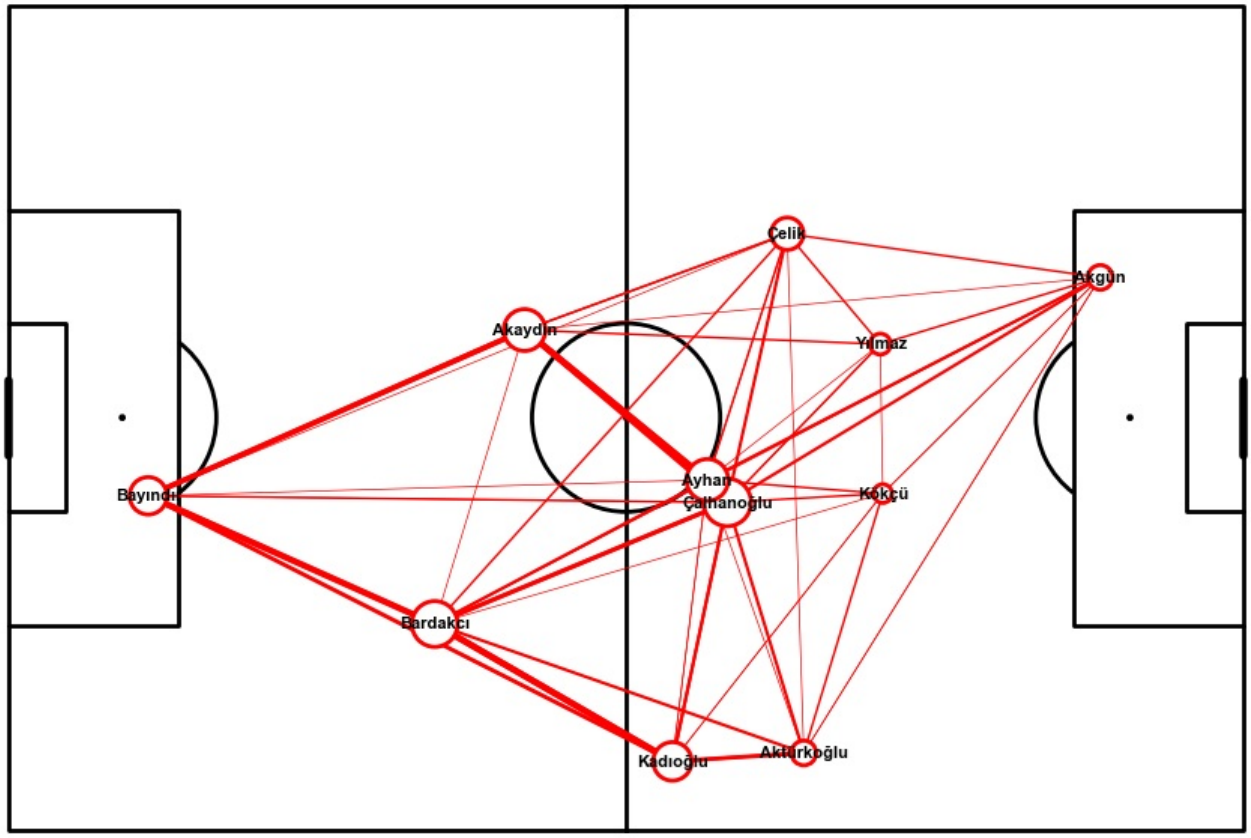
Turkey's Euro 2024 Pass Networks Against Georgia



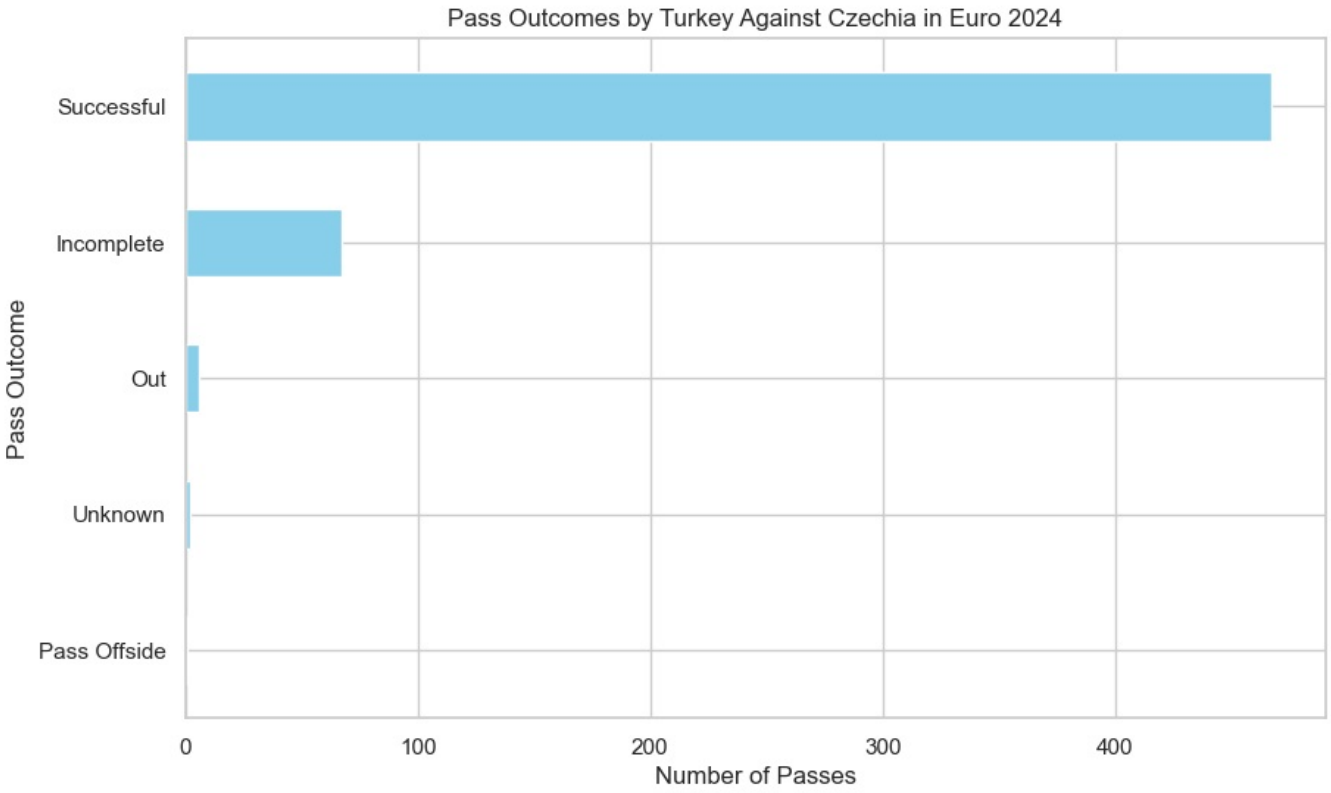
First substitution occurred at 78'



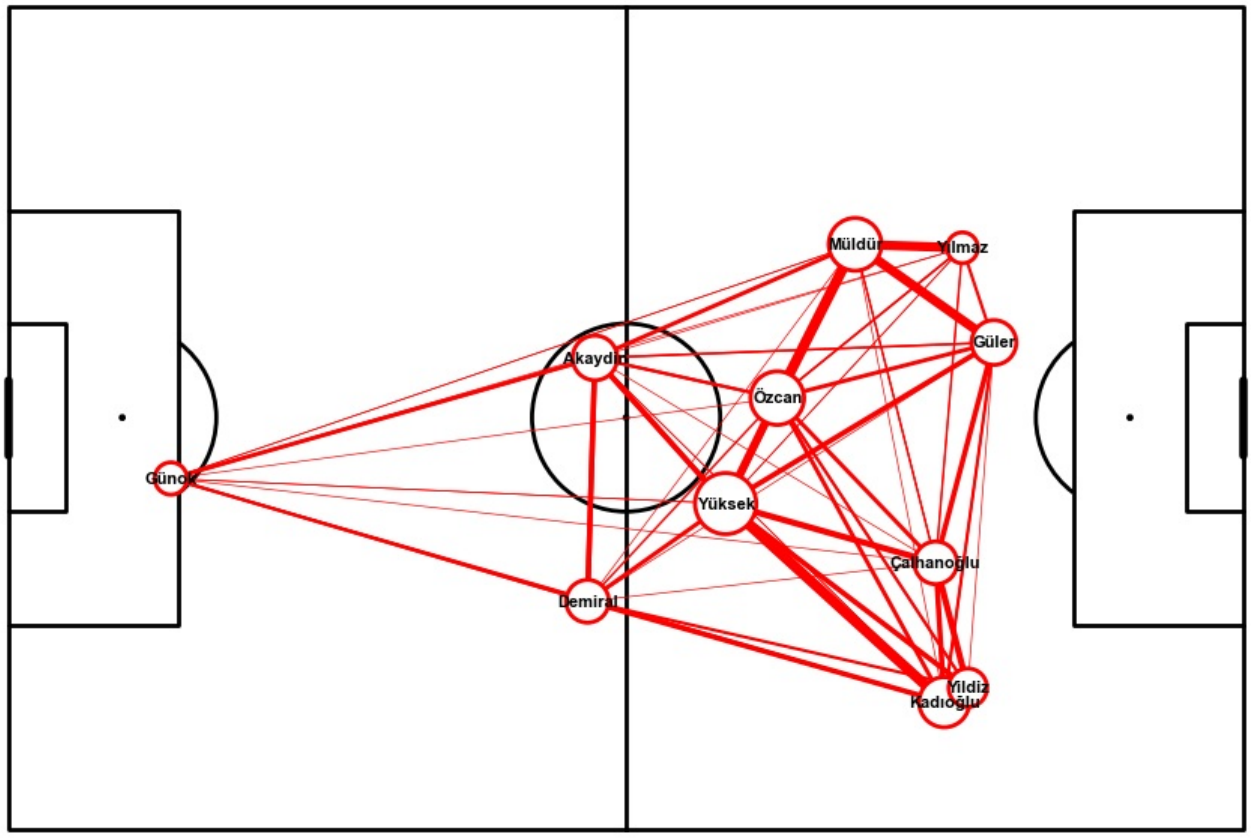
Turkey's Euro 2024 Pass Networks Against Portugal



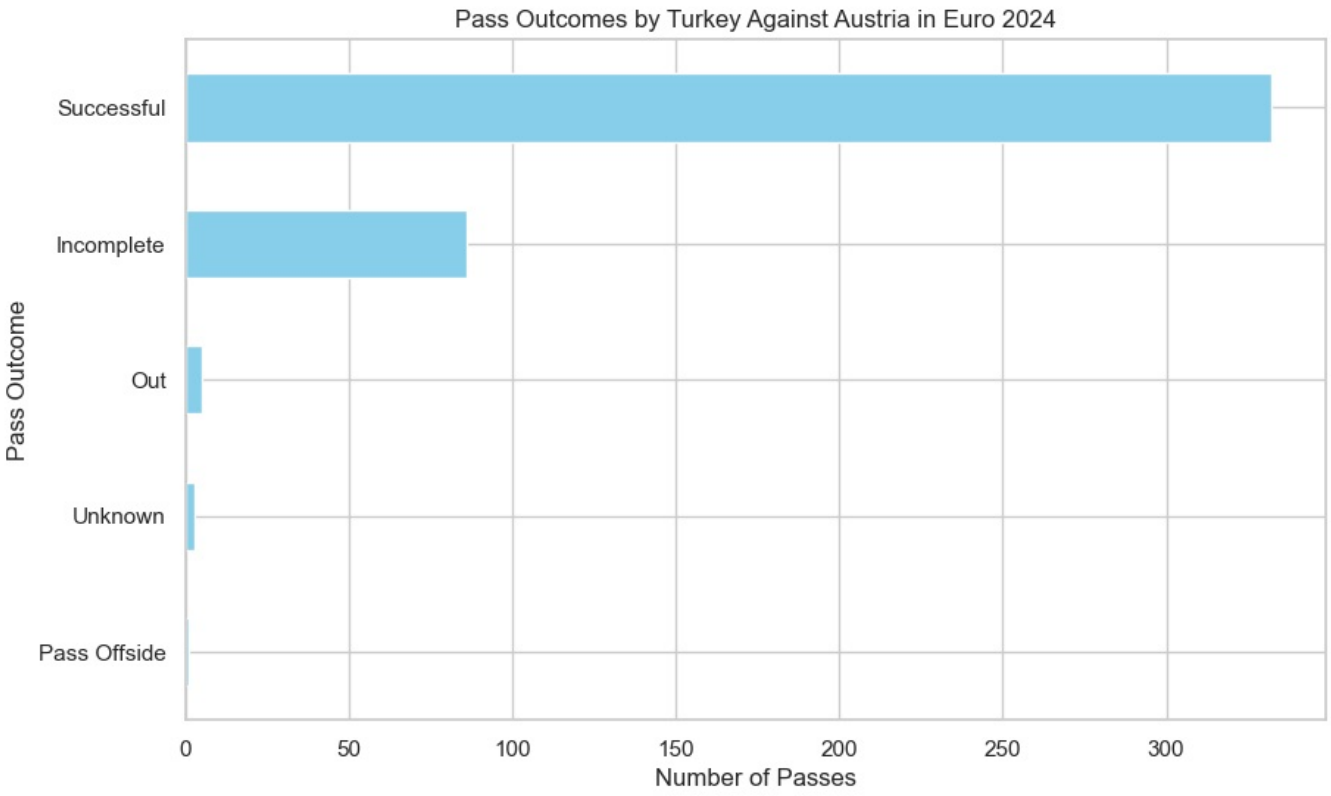
First substitution occurred at 45'



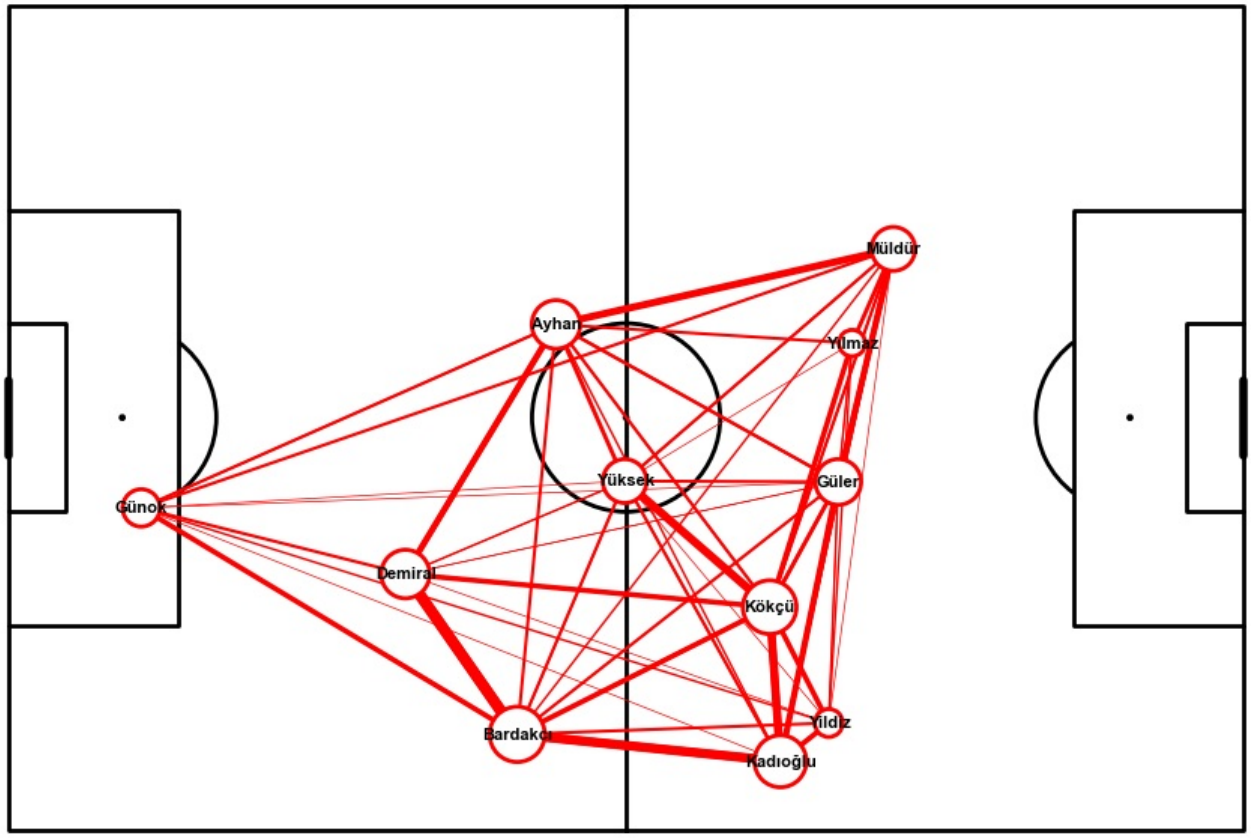
Turkey's Euro 2024 Pass Networks Against Czechia



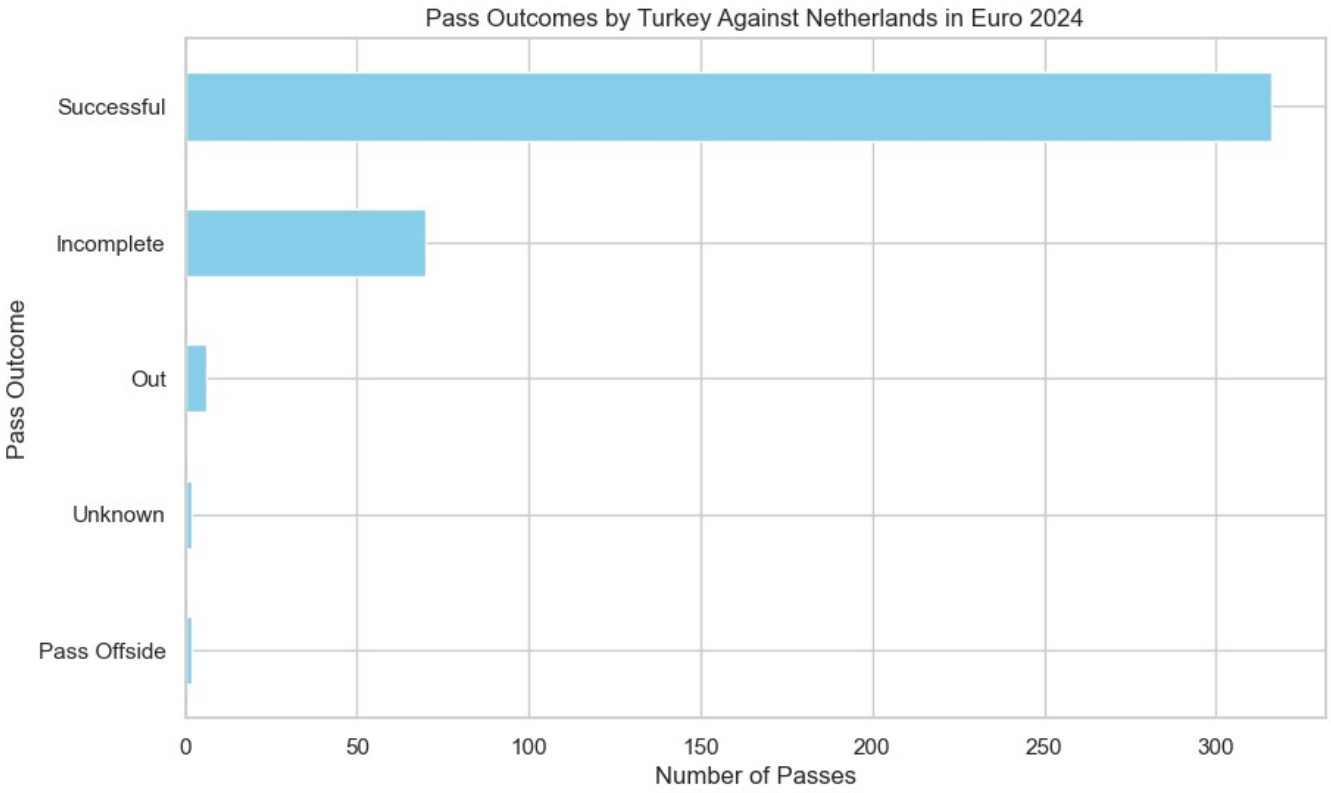
First substitution occurred at 45'

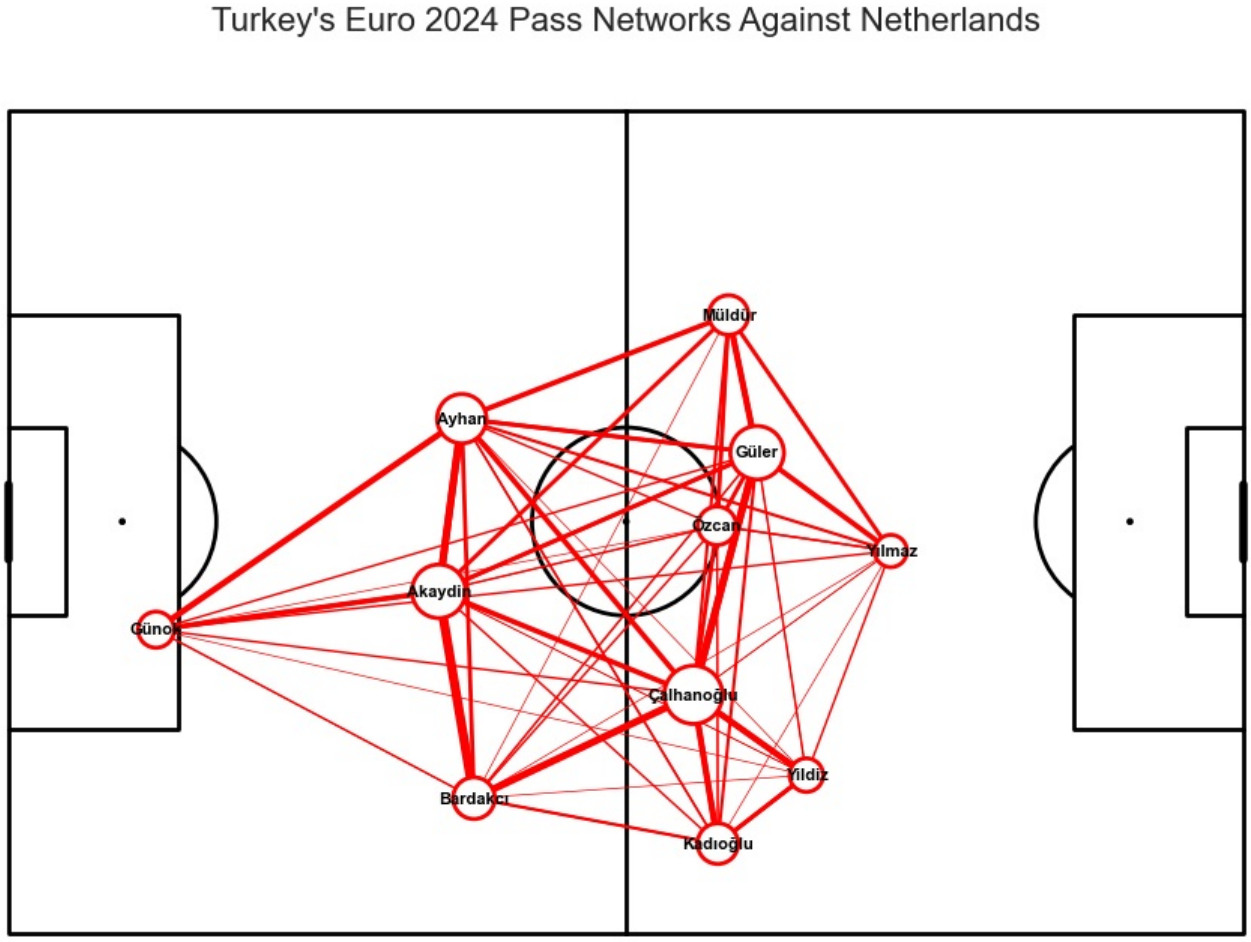


Turkey's Euro 2024 Pass Networks Against Austria



First substitution occurred at 57'





First substitution occurred at 76'

In [ ]: