Spatial representation of skill feasibility for decision making

Applied to robotic football

Master's Thesis
Department of Mechanical Engineering
Control System Technology Group
Eindhoven University of Technology

J.P. van der Stoel[1],
Thesis supervisors:
Hao Liang Chen[1], and René van de Molengraft[1]

[1]Eindhoven University of Technology, Department of Mechanical Engineering, Control Systems Technology Group, 5612 AE Eindhoven, The Netherlands, https://www.tue.nl/en/research/research-groups/control-systems-technology/

January 15, 2023

Abstract

This thesis presents a method 'Semantic Map Decision Making (SMDM)' for formulating the feasible design space of an optimization problem in a spatial manner. Skills and constraints are represented as regions onto the a 2D representation of the environment. These regions compose the feasible space and as a consequence, the distinct between feasibility and quality yields a decrease in complexity of the optimization with respect to most current methods. The SMDM method is developed for a distributed multi-agent robotic football system. A complementary decision algorithm is designed in the form of a Finite State Machine with guarding risk and reward objectives to simulate results real time. It is tested against the benchmark method that uses artificial potential fields. This shows the potential for increasing the configurability of the strategy deployment, since less objectives are needed.

Keywords: Semantic Map Decision Making, Multi objective optimization, Semantic mapping, Spatial representation of constraints, Middle Sized League, Robocup.

Code repository: https://github.com/vanderStoelJaap/SMDM

1 Code of conduct

"This report was made in accordance with the TU/e Code of Scientific Conduct for the Master thesis"

J.P. van der Stoel
0967407
Eindhoven, January 15, 2023

2 Introduction

The increasing exposure of robots to dynamic environments increases the need for flexible decision making to fulfill its task. The proposed type or execution of a skill might change over time with respect to these dynamics. Here a skill is a capability of a robot that allows it to fulfill its task, e.g. 'drive' to complete the task of getting to its destination. In the context of mobile robots this is generally solved by an optimization over some objective. To this end the environment is often discretized by means either grid cells [1] or a vector field [2]. The optimization function then calculates the optimal path or target. In extension, if multiple skills are suited to fulfil the defined task, their optimal solutions are compared to choose the preferred skill. In many situations this optimization is subject to a number of constraints, where for mobile robots those are often spatial relations between features. Spaces could be occupied or a feature is blocking the space behind it. However, practice shows that these constraints are often formulated as optimization objectives. Hence, since constraints and objectives are formulated equally, no distinct is made between feasibility (discrete) and quality (continuous) of the solution. Discrete meaning that a candidate for a solution is either possible or not. It makes more sense to first determine the set of feasible options and then choose the one of highest quality. This reduces either the set of options and the amount of optimization objectives. Which results in more insight in the decision making, improving the configurability.

To demonstrate the effect of the division between feasibility and quality in the decision making, robotic football is taken as an example where the aim is to improve the configurability of the strategy. The study is conducted in close collaboration with Tech United [1] at the University of Eindhoven. This team participates in the Middle Sized League (MSL) of Robocup [2]. In the game of football, strategy has a significant contribution to winning games. The effectiveness of a strategy is dependent on the capability of the players and the opposing strategy, it is therefore something that is adjusted often. In human soccer this is developed over the years and comprises the formation, the type of build up and many more qualitative notions [3]. In the context of MSL, the strategy and configuration of this has been solved in various ways [4, 5, 6]. Currently the Tech United team uses Skills, Tactics and Plays (STP) as the overall strategy framework [7][8]. Within STP each player is given a role and corresponding task. In case the task is 'advance the game', this can be fulfilled by multiple skills (e.g. shoot, pass, dribble) and thus require decision-making. This is solved by the use of artificial potential fields (named 'mu-fields'), computing the optimal skill and target over the gridded field, typically regarding 10 to 20 objectives. Yet, most of these actually describe constraints.

In this work the feasibility of skills are incorporated in the inner model of the robot's environment, represented by semantic 2D regions. The semantic of the regions being the representation of a certain skill. These regions are the

[1] `https://www.techunited.nl/nl/voetbalrobots`
[2] `https://msl.robocup.org/`

result of the defined constraints. The constraints arise from the capabilities of the system or rules according to the game. Consequently, a set of feasible skills, the feasible design space [9], will be available. Instead of optimizing over the positions within these regions, per region a target that is guaranteed to be contained by it is selected. The size of the regions is deemed sufficiently small to allow for this. A finite state machine (FSM) decision algorithm is designed in combination with a number of reward and risk objectives (4). This is tested in simulated environment against a benchmark, the current mu-field implementation of Tech United, to determine if the implementation provides similar behavior while being more configurable.

3 Related work

To familiarize the reader with the topic, Multi-Robot Systems (MRS) are briefly discussed. To highlight the distinction between task allocation and decision making the former is discussed as well as the specific implementation within Tech United. The main contribution off this work is related to the decision making of an individual robot in a 2D environment. Therefore, decision making in general and methods applied in robotic football are evaluated. In this work the decision making aspect is influenced via a semantic map and therefore relevant literature related to semantic mapping is also reviewed.

3.1 Multi-Robot Systems

A MRS is a group of multiple robots trying to perform a long term goal [10]. If the agents have the same capabilities the system is called homogeneous, if they differ heterogeneous. When agents work together to accomplish the common goal the system is cooperative, and if they compete amongst themselves to satisfy their own interest the system is competitive. To execute the long term goal different sub-goals are defined (task decomposition). The allocation of these tasks to the individual agents is the planning problem. Tasks can require one (loosely coupled) or more (tightly coupled) agents. Communication between agents is needed for this coordination, this can be explicit or implicit. Explicit communication uses communication hardware allowing agents to broadcast and receive intentional messages to enhance cooperation. In case of implicit communication agents model (and predict) the behavior of their peers, thereby estimating their intentions. Robotic soccer can be classified as homogeneous system. Although the goalkeeper has different abilities, it never executes tasks of field players and visa versa. Considering a soccer team the system is cooperative, but with regard to the adversarial team it is competitive. Since tasks require only one robot, the task are loosely coupled. There are however dependencies within some tasks, a pass for instant requires another robot to receive it.

3.2 Multi-Robot Task Allocation

An important topic within MRS is the planning problem, named multi-robot task allocation (MRTA) [11]. A task being an objective that could be achieved by executing one or a multiple of skills. A distinction is being made between static (offline) and dynamic (online) planning. The latter allocates every agent with a specific task given the state of the environment and keeps doing this over time (iteratively or continuously). This could be solved in a centralized or decentralized manner, where agents negotiate about the optimal allocation in a market-, action- or trade-based fashion [12].

3.2.1 MRTA on Turtle

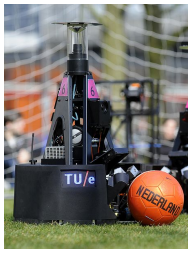

Figure 1: Turtle football robot

Turtle [3] is the football robot in use by Tech United. It is a distributed multi robot system with a shared world model in which the following features are recognized: ball, peers, opponents, goals and outline of the field. The Skills, Tactics and Plays framework [7][8] is used to deploy team coordination. It consists of 'skills' accountable for the execution of the low-level robot behaviour, such as a pass. The 'tactics' are a goal-oriented plan consisting of subsequent actions for an individual robot and are synchronized amongst the team, thus actions are executed sequentially. An action comprises one or more skills. A 'play' is a predefined team plan which assigns roles to each player where each role has specific tactics. A play is chosen based up on the situation on the field. To this end pre conditions and invariants are defined, responsible for entering and leaving a play respectively. Once a play is selected, the roles defined by the play must be fulfilled by the different robots. Roles are assigned by minimizing the distance between peers and roles. To ensure a global agreement the robots share their cost matrix. Each robot then computes the aggregate cost matrix, thereby having a global optimal solution. During the play the roles are still assessed to enable switching from roles within the play. In case of optimization within an action, a decision is made trough the mentioned artificial potential fields (visualized in figure 3).

[3] A comprehensive hardware description of Turtle can be found at `http://roboticopenplatform.org/wiki/TURTLE`

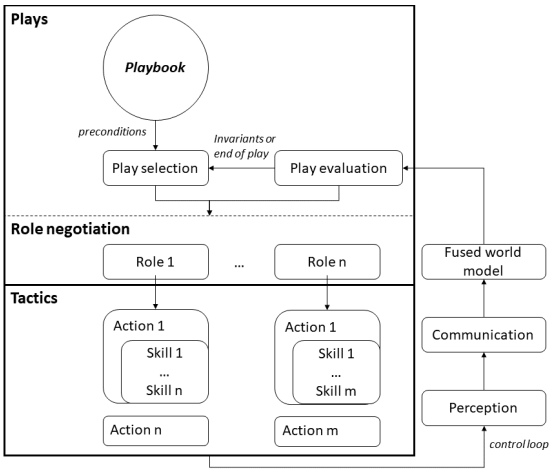

Figure 2: STP framework

3.3 Decision making

Actions within a task are not always deterministic. In many cases the target or path of the skill depends on the state of the environment, and should be decided on ad-hoc. For navigation this is typically done via a path planning algorithm [13]. In cases the appropriate skill within an action can not be determined a-priori, often this also depends on the future state of the environment, decision making is also required. Naturally this decision making relies on multiple (contradicting) objectives. Literature provides a variety of methods [9]. The general expression of a multi objective optimization is shown in equation 1.

$$\begin{aligned} \underset{x}{\text{Minimize}} \quad & f(x) = [f_1(x), f_2(x), ..., f_k]^T \\ \text{subject to} \quad & g_j(x) \leq 0, j = 1, 2, ..., m \\ & h_i(x) = 0, l = 1, 2, ..., e \end{aligned} \tag{1}$$

where k is the number of objective functions, m is the number of inequality constraints, and e the number of equality constraints, $x \in E^n$ is a vector of design variables (also called decision variables), where n is the number of independent variables x_i. $f(x) \in E^k$ is a vector of objective functions $f_i(x) : E^n \to E^1$. $f_i(x)$ are also called objectives, criteria, payoff functions, cost function or value functions. The feasible design space $\mathbf{X}$ (often called the feasible decision space or constraint set) is defined as the set $\{x \,|\, g_j(x) \leq 0,\ j = 1, 2, ..., m;$ and $\ h_i(x) = 0,\ i = 1, 2, ..., e\}$. The feasible criterion space $\mathbf{Z}$ (also called the feasible cost space or the attainable set) is defined as the set $\{f(x) \,|\, x \in \mathbf{X}\}$. Feasibility implies that no constraint is violated.

3.3.1 Decision making on Turtle

Tech United currently uses potential fields to determine which skill to select (shot, pass, dribble) and the optimal target position of it, this is similar to the utility-map created in [6]. Both of them correspond to the weighted weighed sum utility function of equation 2 [9], where the decision variables are the positions (x, y) on the field.

$$U = \sum_{i=1}^{k} w_i f_i(x) \tag{2}$$

Here w_i are the weights of the individual objectives f_i. With respect to attacking play two types of mu-fields (potential-fields), one for choosing position and one for dribbling are defined. Their objectives are depicted in keywords in table 3 of appendix 7.1. Players without the ball use the first to determine there target position and communicate this to the player in possession as pass target. The latter is used by the player in possession of the ball to determine the dribble target for a shot, if the target is equal to its current position. Then the player in possession compares the costs of these targets and performs the corresponding skill. However in case a shot is possible, the decision will be overruled and a shot is performed. As already stated in the introduction constraints on skills are formulated as optimization objectives with a relative high weight. Therefor $\mathbf{X}$ is not confined, rather all positions on the field are evaluated. A typical constraint for a pass would be the space behind an opponent, this is then introduced as an objective. When a position behind an opponent is evaluated, a high cost relative to other objectives is introduced to make sure this position is not selected. In this work, by explicitly incorporating the feasibility of the allowed skills in the world model, $\mathbf{X}$ can be determined prior to the optimization in the form of 2D regions,this allows for a distinct between feasibility and quality.

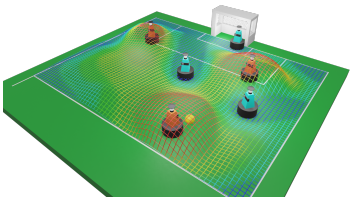

Figure 3: Visualization of mu-field used for decision making

3.4 Explicit mapping trough semantics

Adding semantic knowledge to the robots world model adds a layer of abstraction that enables more complex reasoning [14]. Trough the semantic knowledge spatial relations between different features in the environment can be described that would be hard to communicate otherwise [15]. There exist an established

foundation for the creation [16] and storage [17] of these semantic knowledge models. Hao Liang Chen et al. [**?**] presents a method to solve the motion control of a mobile robot by a semantic map. Constraints and objective functions are related to the semantic areas in the world, where the Constraint-based Optimization Problem (COP) is generated based on the semantic areas in which the robot resides. In this work the semantics of a skill will be translated to a 2D region on the world model of the robot. These regions are confined by the constraints following from the capabilities of the system or the rules set by the environment, in this case the game of football.

4 Methodology

The essence of this work is a method, from now on refereed to as 'Semantic Map Decision Making' (SMDM), to create a semantic 2D representation of the feasible design space for a certain skill of a robot. This chapter introduces the general formulation and a visualization of the result, in the next chapter the implementation for the specific use case is given. Assumed is that a (mobile-) robot is set, e.g. by a task planner, to fulfill a certain task. This task can be performed by i different type of skills S_i. Here S_i describes a 2D region which can either make up the whole environment or, as in this work, a predefined region within the environment. For example, all positions that a receiver can attain within a reasonable amount of time, describe the region of the skill 'pass'. The decision variables, in this case being positions $(x, y) \in \mathbb{R}^2$, are subject to a set of constraints, similar to (in-)qualities, except now defined by a 2D region C_i. Constraints could describe a relation between a skill and a certain feature. E.g. a football robot cannot give a pass to the space behind another player. Thus a spatial representation of these features F_i is needed. Here, a feature is an physical object in the environment and i describes the type of feature. Now $N(S_i)$ is the set of constraints relevant for skill S_i, then the difference between a skill region and its confining constrained regions $(S_i - \bigcup_{i \in N(S_i)} C_i)$ compose the feasible design space $\mathbf{X}$. From now on this is referred to as the feasible space $\mathbf{O}$ as defined in equation 3. Note that every skill has its own feasible space $\mathbf{O}$.

$$\mathbf{0}(S_i) = S_i - \bigcup_{i \in N(S_i)} C_i \tag{3}$$

In figure 4 the regions are visualised for a typical situation in robotic football. The specific implementation and mathematical description leading to these regions are described in the next chapter. Three peers, of which one in possession of the ball, and three opponents are shown. Their positions affects the spatial relations between them, and thus the feasible space. E.g. for the peer on the left it is clear that the region in which it is able to receive the ball is affected by the opponent in front. Since it is not able to receive the ball behind the opponent.

Assuming constraints are defined correctly, such that they represent the restrictions of the system in the real world, the feasible space guarantees the

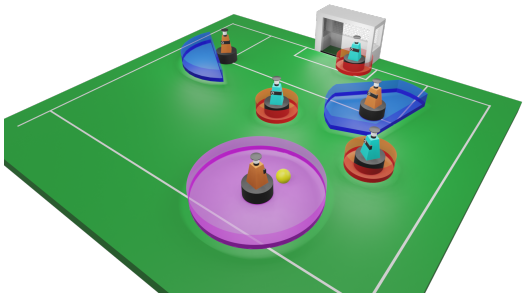

Figure 4: Feasible space **O** displayed by semantic regions (pass, dribble, shot)

feasibility of the evaluated decision variables. In the particular case of robotic football those are position on the field, $(x, y) \in \mathbb{R}^2$. This allows for less complex decision making, as advocated before, of which an implemented example is given in the following chapter.

5 Implementation

The methodology is applied to the specific case of robotic football. To this end a module is written in Python by using Shapely [4]. This module has as input a set of features and computes the semantic 2D regions describing the feasible design space. It also comprises a decision algorithm in the form of a Finite State Machine to select the appropriate skill and returns this. For integration with the software of Turtle [5] a Python/C API [6] is utilised. Shapely is a package for deterministic spatial analysis and comprises the implementation of the set theory[18] concepts needed to create the semantic regions that will make up the feasible design space, as described in the methodology. The software architecture of the module is depicted in figure 5 and used settings used in simulation are presented in appendix D.

[4] Shapely manual `https://shapely.readthedocs.io/en/stable/manual.html`
[5] Turlte software `https://gitlab.tue.nl/tech-united-eindhoven/Turtle3`
[6] API for embedded Python `https://docs.python.org/3/c-api/index.html`

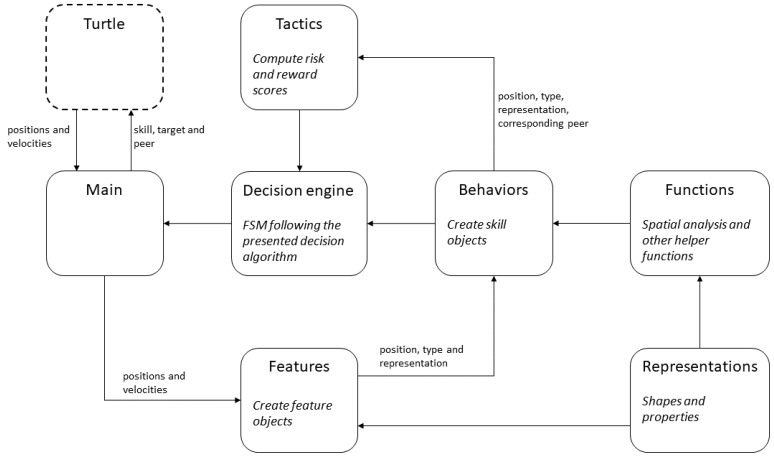

Figure 5: Software architecture of the implemented SMDM module

Currently the module only accounts for the decision making of the player in possession. This means that players off the ball still use the mu-fields for determining there optimal position. The module is used to simulate the method of which the results are presented in the next chapter.

5.1 Feasible space

To create the feasible space, the different features involved need to be known in order to apply the constraints that are based upon them.

$$F_i \mid i \in 1, 2, 3, 4, 5 = \textit{field, goals, ball, peers, opponents}$$

Furthermore the peers F_4 and opponents F_5 have an avoid region $A(F_i)$ to prevent collision and in case of opponents to prevent interception of a pass or shot. This avoid region is an expansion of their area and stretches in the direction of their velocity to form an ellipse, as described by equation 4 with a constant $r = 0.8[m]$ for peers and $r = 1.2[m]$ for opponents.

5.1.1 Skills

The initial representation of the skills of turtle are defined, note that only skills used in attacking plays are regarded.

$$S_i \mid i \in 1, 2, 3 = \textit{shot, pass, dribble}$$

A shot is initially defined to the whole environment and will be constraint by the opponent goal, as described in the following chapter. A dribble is defined as a circle with a radius of $3[m]$ with a fixed center on the position where the

ball was received. The radius is according to the rules of the game [7]. A pass is represented by an ellipse around the peer of interest. The position of the peer is one of the focal points and the other one is in the direction of the velocity of the peer. The minor and major axis of the ellipse are computed according to equation 4.

$$\begin{aligned} minor\ axis &= r \\ major\ axis &= r + f_v * v_t \end{aligned} \tag{4}$$

where r is the radius following from equation 5, v_t the velocity of the peer and f_v the velocity factor. This factor determines the size of the feasible space and is iteratively determined at $f_v = 0.5$, by simulating the effect. This value results in a reasonable area for the pass skill, where it does not exclude positions that seem attainable.

$$r = \begin{cases} f_v * v_t * \frac{d}{v_p}, & \text{for } r > r_{min} \\ r_{min}, & \text{otherwise} \end{cases} \tag{5}$$

Here v_p is the pass velocity which is assumed to be constant with $5[m/s]$, r_{min} is the minimal radius which is set at $2[m]$.

5.1.2 Constraints

Every skill has a set of constraints confining the feasible design space. These spatial relations are explained in table 1.

Table 1: Set of spatial constraints for robotic football

Skill	**Constraint**	
S_1	C_1	All spaces not onto the goal
	C_2	Spaces behind opponents
S_2	C_3	All spaces outside the field
	C_2	
	C_4	Spaces closer to an opponent than the receiver
S_3	C_5	Spaces directed towards opponents

The first constraint C_1 restricts the feasible space S_1 of the shooting sill to the area of the opponent goal (dark red region) in figure 6. In order to score, a shot should always be aimed at the opponent's goal.

$$C_1 = F_2 \tag{6}$$

[7] `https://msl.robocup.org/rules`

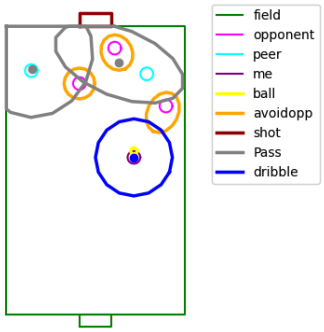

Figure 6: Feasible space as a result of C_1, C_3

The second constraint does not allow positions behind an other player. Turtle is not capable of passing over another player therefore these regions are not within the feasible design space. The region of this constraint is defined by the scaled convex hull $sc\_Conv(F_3, A(F_i^n))$ of the ball and the avoid region of the opponent, minus the difference of the non-scaled convex hull and the actual avoid region of the opponent. Hence, only spaces within the avoid region and behind it are constraint. This arbitrary scaling of 10 times ensures all positions behind the other player to be within the constraint region. The procedure is described by equation 7, where n is the number of features of type i. This is visualized in figure 7, where the constraint regions are annotated in red. A description of the convex hull is given in appendix C.

$$C_2 = \sum_{1}^{n} sc\_Conv(F_3, A(F_i^n) - ((Conv(F_3, A(F_i^n)) - A(F_i^n)) \tag{7}$$

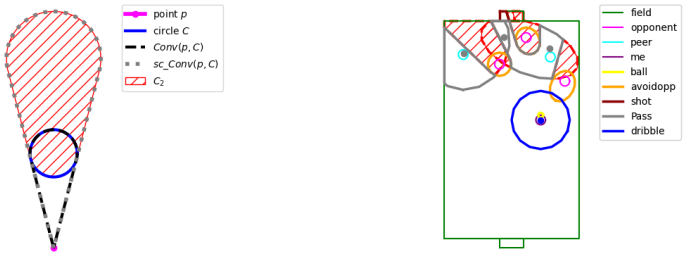

Figure 7: Feasible space as a result of C_2

The third constraint bounds the skill space S_i by the lay out of the field, shown in figure 6 where the feasible spaces do not intersect with the outline of the field. As set by the rules of (robotic) soccer a player can not receive or dribble a ball outside of the field.

$$C_3 = F_1 \tag{8}$$

The fourth constraint excludes positions that are closer to any opponent then to the peer to which a pass is performed. A position closer to an opponent allows the opponent to intercept the ball before the peer is able to reach the ball. Hence a Voronoi diagram is created based upon the positions of the opponents and peers, this is visualised in figure 8 (left), where the point in the middle of the circle is considered the point of interest. The constraint space is then made up by the difference between the feasible space and the Voronoi cells that intersect this space but do not contain the point of interest, and are annotated in red in figure 8 (right). Equation 9 formulates this spatial relation. Note that the velocity of the features is not taken into account here, instead only their position is used to create the diagram. A description of the Voronoi diagram is given in appendix C.

$$C_4 = S_i - (S_i \cap voronoi(\sum_{1}^{n} A(F_{5,n}))) \tag{9}$$

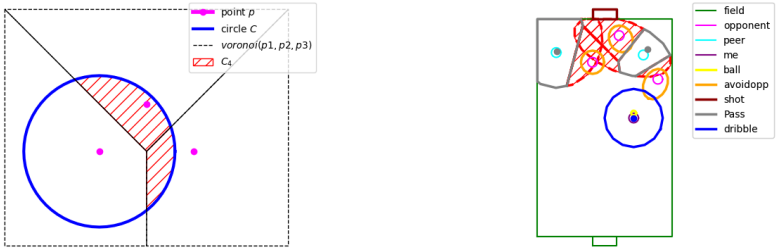

Figure 8: Feasible space as a result of C_4

The fifth constraint excludes positions that are directed towards an opponent. A dribble in the game of football is used to dodge an opponent and create a new situation with opportunities for a successive skill. To determine those spaces the dribble region is split up between a region directed towards the goal and a region directed away from the goal. Then the convex hull of the ball and opponent is taken and its intersection with the feasible space defines the constraint space, equation 10. Because of the mentioned split towards and away from the goal this creates multiple feasible spaces with there own target position

to optimize over in the decision making. This is shown in figure 9.

$$C_5 = \sum_{1}^{n} sc_C onv(F_3, A(F_{5,n})) \tag{10}$$

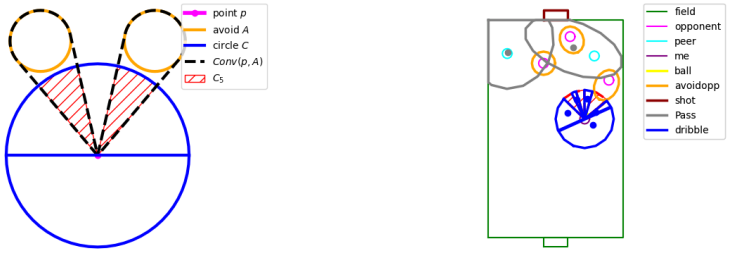

Figure 9: Feasible space as a result of C_5

The resulting feasible design space then follows from combining these constraints as in equation 3. This is shown in figure 10, here every point within the defined regions is guaranteed to adhere to the constraints.

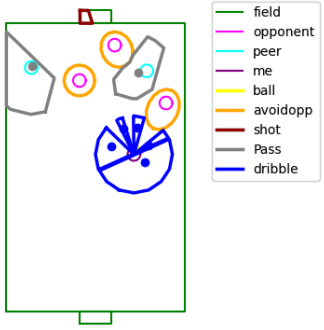

Figure 10: Representation of feasible space for an arbitrary situation

5.2 Evaluation points

In the aftermath of the feasible spaces, since these do provide a confined region, it is deemed sufficient to generalize this region by one point contained

by it. Assumed is that this still leads to desired behavior, it is however evident that the solution will in most cases be sub-optimal to an optimization over all positions in the region. For a shot the target position is not used, since the Turtle software determines the exact location in 3D and thereby able to vary the height. For a dribble the target is computed by the Shapely function `representative_point()`. The target of a pass in placed in the direction of and based upon the receiver's velocity if this is within the feasible design space, else the representative point is calculated. This is depicted in equation 11. The representative point is a cheaply computed point within a region, by use of Shapely.

$$p = \begin{cases} \frac{1}{2} * (\frac{v_{t,i}^2}{a_{max}+rt)}, & \text{for } \{i = x, y | p \in \mathbf{O}\} \\ \texttt{representative\_point()}, & \text{otherwise} \end{cases} \tag{11}$$

Where p is the target position, v_t the velocity of the receiving peer, a_{max} the maximum acceleration $4.5[m/s]$ and rt the reaction time of $\frac{1}{samplerate}$.

5.3 Decision making

Given the feasible design spaces of the different skills and the design variables, one target per feasible space, a decision can be made between the possible skills. In comparison to the currently used mu-fields this relies on less objectives as well as fewer design variables because of the derived feasible spaces. The latter even more so since one target point per feasible space is evaluated. There is chosen for a finite state machine (FSM) with guarding thresholds on risk and reward, this is displayed in algorithm 1. It is not claimed that this algorithm performs better than current methods, rather it is used to display the effects the semantic regions confining the feasible design space.

5.3.1 Objectives

The decision between the different possible skills in a game situation are based upon two risk and two reward parameters. The selection of them is based upon the current objectives used in the mu-fields and prior knowledge about the game [19][20], partly from the futsal soccer manual [3]. The following risk parameters are defined:

1. distance to the target position
2. the area of the feasible space as a measure of the pressure on the ball

The distance between the current position and the target position is a measure for the duration to complete the skill. A bigger distance will give an opposing player more time to react, this increases the change of the opponent intercepting the ball. Moreover, this gives the opposing team the time to reposition and hence an initially favorable decision could actually be a poor option once the skill is completed. The area of the skill is considered as a quantity for

the pressure on the ball [21]. A small area of the feasible space directly relates to the presence of one or more opponents in most cases. Opponents that are close to the player in possession form a direct threat for losing the ball. Only in the situation when performing a dribble and the player gets close to the border of this feasible space, because of the division between directed from and towards the goal, this risk parameter does not relate to the pressure on the ball. Currently these two parameters are only used to decide if a feasible space is valid, in case it exceeds the threshold the skill is removed. The decision between valid options is based on two reward parameters:

1. opponents between the ball and the goal
2. distance to the penalty spot

Here the amount of opponents is a measure for the likelihood to reach the goal and perform a shot. Fewer opponents between the ball and the goal increases the chance of scoring, thus results in a higher reward. Furthermore the penalty spot is assumed to be the ideal position to score from. Therefor the distance to this point quantifies the quality of the position, a lower distance results in a higher reward.

According to the existing knowledge about the game [3] there is a preference between skills. A shot is the only skill that could directly lead to a goal, furthermore the general rule is 'pass if you can, dribble if you must'. Hence the skills are evaluated in that order and if its reward is above the threshold the skill is performed without regarding its alternatives. In contrast, if its risk is above the threshold the skill is removed as an option. In most cases no skill adheres to the reward threshold. Then skills are sorted based upon their reward scores and the one with the highest score is selected, therefor the skill with the best reward according to the objectives is chosen. Different to the mu-fields, not all parameters are implemented as a continuous value. The amount of opponents between the ball and goal is a discrete number. Therefor this is sorted first and sequentially the combined score of the other rewards (global sum) is regarded, in this case only the distance to the penalty spot.

Algorithm 1 FSM decision algorithm used to validate the SMDM method

```
function StateMachine(threshold settings)
    shots = Behavior.shot()              ▷ feasible spaces of shots are computed
    for shot in shots do
        if shot.reward ≥ shot_reward_threshold then
            return shot
        end if
    end for
    passes = Behavior.pass()             ▷ feasible spaces of passes are computed
    for pass in passes do
        if pass.risk ≥ pass_risk_threshold then
            remove pass
        end if
        if pass.reward ≥ pass_reward_threshold then
            return pass
        end if
    end for
    dribbles = Behavior.dribble()  ▷ feasible spaces of dribbles are computed
    for dribble in dribbles do
        if dribble.risk ≥ dribble_risk_threshold then
            remove dribble
        end if
        if skill.reward ≥ dribble_reward_threshold then
            return dribble
        end if
    end for
    skills = [shots, passes, dribbles]
    skill = sortSkills(skills, reward)        ▷ sort skills based upon objectives
    return skill
end function
```

6 Results

The implemented method is tested real-time in the Tech United simulator and an animation is created [8], with filename *'ANIMATION-1(10-01-23).mp4'*. Two particular situations of the proposed method are compared to the benchmark mu-fields. In addition quantitative results about the feasible design space, number of decision variables and computational effort are given.

Figure 11 shows a snapshot of the animation. Here the Tech United simulator is shown together with the composed semantic map. The situation shows 2 peers and 3 opponents (circles), and in the upper right corner the corresponding the feasible spaces can be observed. It is not clearly visible on the picture, but the peers are numbered as 2 (light blue border and in possession of the ball)

[8] `https://github.com/vanderStoelJaap/SMDM/tree/main/animation`

and 3 (the assisting player). Player 2 is in possession of the ball in the right forward position and directed towards the goal. Player 3 assists this player but is positioned behind an opponent with respect to player 2. This results in the feasible space being split up in a region left and right of this opponent. Player 2 does also have the option to dribble, where the feasible space is divided into 4 regions which avoid the opposing players. It also has the option of a shot at goal, as observed by the presence of the dark red feasible space related to the shot skill. According to the decision making procedure of algorithm 1, the best action in this situation was to shoot at the goal.

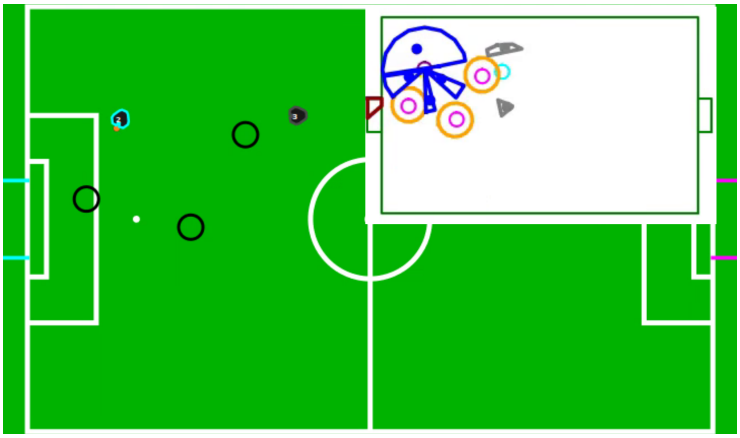

Figure 11: Situation 1: snapshot of simulation and corresponding semantic map at $t = 1.22[s]$

This scenario is also simulated using the mu-fields and depicted in figure 12. Since the mu-fields are calculated decentralized, according to the objectives defined in appendix A these are visualised for the player in possession of the ball on the left and for the assisting player on the right. The color scaling correspond to a high cost (light blue) and low cost (dark red) of the optimization function for every grid cell. The white cross represents the position with optimal score for each player. For the assisting player (3) it can be observed that indeed spaces behind opponents, from perspective of the pass giver (player 2), do have a high cost and are not favored. This corresponds to the pass regions created, the difference being that in this case it is taken into account as optimization objective. The pass target however is placed near the left forward position. The large distance to this target follows from the fact that the feasible space is not restricted. The mu-field player in possession of the ball (2) is only calculated over the allowed dribble region and shows a high cost for positions towards opponents, similar to the dribble regions. Furthermore the cost increases with the distance to the target. Player 2 is already at his dribble target and thus a shot could be performed. Since this is favored above a pass to player 3 a shot was taken in this situation.

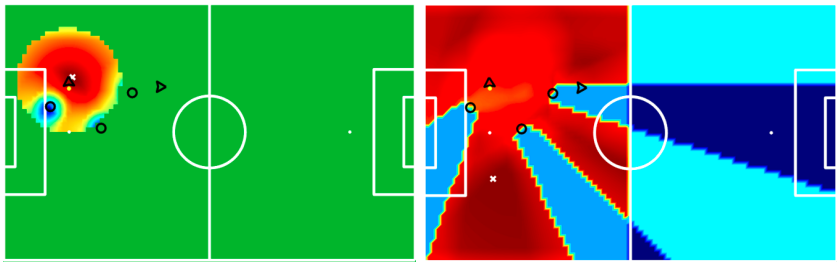

Figure 12: Mu-fields corresponding to situation 1 for the player in possession (left) and assisting player (right)

In the second situation as shown in figure 13 two opponents block a possible shot on goal. A pass to the assisting player (3) is possible, however confined by the constraint (C_4) that every position in this region should be closer to the assisting player than any opponent. Alternatively three dribbles are possible. A pass to the assisting player is performed because it has fewer opponents towards the goal, according to the decision algorithm 1 and related reward objectives.

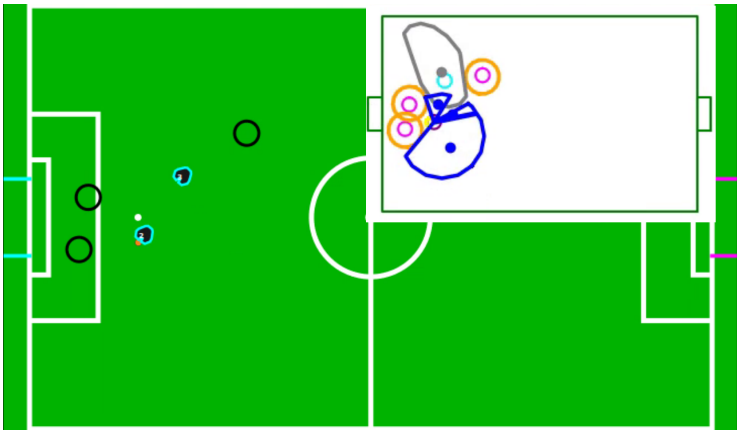

Figure 13: Situation 2: snapshot of simulation and corresponding semantic map at $t = 1.30[s]$

The corresponding simulated mu-fields are given in figure 14. They again show to correspond well with the created feasible space, where constraint spaces in the semantic map align with high cost positions in the mu-fields. In extension, a pass was performed as well using the mu-fields.

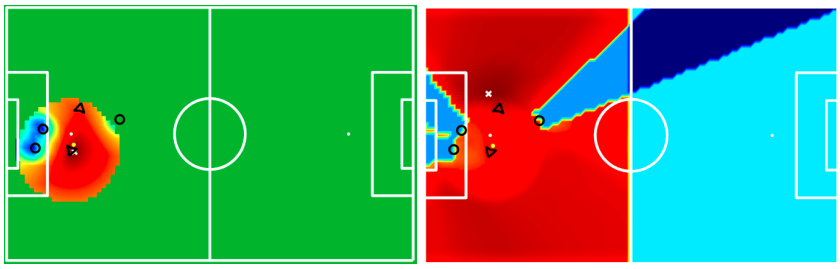

Figure 14: Mu-fields corresponding to situation 2 for the player in possession (left) and assisting player (right)

A quantitative comparison with the mu-fields is given in table 2, here the situation considered in the implementation (figure 10) is taken as a basis. The area of the feasible design space reduces from the whole field to a set feasible spaces, equivalent to a reduction of 77%. Consequently, this leads to the same reduction in evaluation points (grid cells). However since only one evaluation point per feasible space is regarded this reduces even further to only 8 possible solutions. Also the amount of objectives in the optimization reduces from 14 to 4 in case of the mu-field for the assisting player (walk free), and from 9 to 4 in case of the mu-field for the player in possession (advanced attack), the latter is not shown in the table. Note that this mu-field also does have fewer evaluation points as well, since this is only calculated over the allowed dribble region. Lastly, it is clear that the SMDM method takes more computational effort at this point with a maximal sample rate of 25 $[Hz]$ compared to 75 $[Hz]$.

Table 2: Results on the complexity of the decision making

method	**feasible design space**	**evaluation points**	**objectives**	**max sample rate**
mu-field	311.15 $[m^2]$	4978	14	75 $[Hz]$
feasible spaces	72.49 $[m^2]$	8 (1160)	4	25 $[Hz]$

6.1 Discussion

The simulated results show the feasible spaces to be properly defined in every situation according to the defined constraints. This points towards a correct formulation of the constraints. Behavior does show to be insufficient at particular moments. First of all some passes are given behind a peer and to slow, which indicates a seemingly poor chosen pass target. This can either be the result of the pass target calculated in front of the peer is not contained by the feasible space or the derivations of the formulas calculating the pass target need some revising. The latter one seems to be case since this often happens when the peer has a high velocity. Moreover, sometimes the players demonstrate a

passing back and forth while it seems more logical for the player in possession to dribble and wait until the assisting player has moved to a better position. This is highlighted by the fact that opponents are static features in the simulation. Whereas if opponents were dynamic or more assisting players would be present, the environment would change such that another skill would be chosen within a reasonable amount of time. Even tough this behavior appears to be the result of the settings of the risk-reward parameters, indicating that those require some additional tuning.

The two compared situations are not sufficient to state similar performance in terms of behavior compared to the current technique, the mu-fields. Nevertheless regarding the simple state of the game, 2 dynamic players versus 3 static opponents, the selected skill is similar to the mu-fields to a certain extend. Where the variations observed, the difference in pass target in the first situation, can be explained by the different approach defining the initial region representing the skill. The mu-fields evaluate the whole environment and thus a distant pass target was selected in the situation of figure 12. Whereas the SMDM method defines an initial region around the receiving player based on the distance and its velocity, as described by equation 9.

The quantitative results clearly show the decrease in evaluation points and fewer objectives. This is a direct result of the distinction between feasibility and quality in the optimization. In spite of that the implementation does require a substantial larger computational effort. Underlying factors for this are the choice of language (Python) which is not embedded and the fact that the method is implemented in a centralized manner. Therefor the player in possession has to calculate all feasible spaces, whereas it might be beneficial for its peers to compute their particular pass region.

7 Conclusion and future work

The Semantic Map Decision Making (SMDM) is a method that enables the formulation of the feasible design space of an optimization function as 2D regions. This proves to be useful in optimization problems with constraints described by a spatial relation between features, such as in robotic football. An initial skill region can be defined, describing the whole or part of the environment. Dependent on the skill a number of constraints are applied, which are again represented by a 2D region. The composure of the eventual feasible space, follows from these constraints by using set theory.

As a result the optimization function has less evaluation points, since not the entire environment is considered. In addition the amount of objectives is decreased with respect to the current mu-fields, where the constraints are formulated as objectives. Once the skills and constraints are defined, the feasible space emerges naturally from them and assures all points contained by it to be feasible. Another advantage of these feasible spaces is that, if the region is small enough, a single point within it can be used as target position for the skill.

The implementation still requires some work as it is currently only imple-

mented for the player in possession of the ball and not in a distributed fashion. Simulated results show desired behavior in real-time. Performance does not meet the current mu-fields in terms of the observed game play. However, this shows potential for improvement and has not been the objective of the work. Hence making the distinct between feasibility and quality of the evaluated points. The decision algorithm created is by no means optimal and requires further investigation. Although the optimization objectives were carefully chosen based on the extensive existing knowledge about the game. Results verify the method in a real-time simulated environment and show the decrease in complexity of the optimization. Consequently the Semantic Map Decision Making fulfills in the goal of making the strategy more insightful en thereby shows the potential for enhanced configurability.

7.1 Recommendations

This work might extend in a few directions now the potential is proven. Implementing the proposed method for players off the ball to position themselves would result in the ability to perform attacking open play fully based upon the semantic map. This would be the author's choice of preference, attacking situations require pro-activity and thus more complex decision making whereas defending is more reactive.

In extension to this, it would be interesting to incorporate formations. The current STP framework only distinct between intentions, e.g. 'attacker', 'defender'. However for more complex strategies it is beneficial to allocate a role with a certain position, e.g. 'left winger'. The proposed method allows for the implementation of this by defining it as a constraint. Here the size of the region is a measure for the freedom a role has to leave its formation position. Additionally it might be wise to make this region depended on the position of the ball, this is already discussed in other work within the MSL [5]. The reason for this that sometimes the shape of the formation changes with respect to the game state. E.g. when the right forward has the ball around the corner, the left forward is preferred to be in front of the opponent goal to receive a pass.

Furthermore, an improvement to the method is to implemented in a distributed fashion, just as the mu-fields. The player in possession only computing the regions for a shot of dribble, receiving the potential pass target and corresponding cost from its peers. The peers calculating the optimal target for positioning and the target for receiving a pass (if possible), since those will likely share most constraints. By distributing the computing the performance increases. Looking further ahead, learning the weights for the objectives should be seriously considered. Since this method results in fewer objectives this could open the door for learning techniques to configure the optimal strategy. If this is done real-time it will allow the players to alter the strategy based on the opponents reaction.

Lastly, there is still room for improvement of the method. If constraints mimic the real world more, the more likely that the feasible design space correspond as well. Therefor the incorporation of the robot velocity in both the

size of the avoid regions ($A(F_{i,j})$) and the reachable area (C_4) will increase performance. As both are currently computed solely based on the position.

References

[1] H. Moravec and A. Elfes, "High resolution maps from wide angle sonar," in *Proceedings. 1985 IEEE international conference on robotics and automation*, vol. 2, pp. 116–121, IEEE, 1985.

[2] J. Borenstein, Y. Koren, *et al.*, "The vector field histogram-fast obstacle avoidance for mobile robots," *IEEE transactions on robotics and automation*, vol. 7, no. 3, pp. 278–288, 1991.

[3] FIFA, "Futsal coaching manual," June 2019.

[4] E. Antonioni, V. Suriani, F. Riccio, and D. Nardi, "Game strategies for physical robot soccer players: A survey," *IEEE Transactions on Games*, vol. 13, no. 4, pp. 342–357, 2021.

[5] L. P. Reis, N. Lau, and E. C. Oliveira, "Situation based strategic positioning for coordinating a team of homogeneous agents," in *Balancing Reactivity and Social Deliberation in Multi-Agent Systems*, (Berlin, Heidelberg), pp. 175–197, Springer Berlin Heidelberg, 2001.

[6] A. J. R. Neves, F. Amaral, R. Dias, J. Silva, and N. Lau, "A new approach for dynamic strategic positioning in robocup middle-size league," in *Progress in Artificial Intelligence* (F. Pereira, P. Machado, E. Costa, and A. Cardoso, eds.), (Cham), pp. 433–444, Springer International Publishing, 2015.

[7] L. de Koning, J. P. Mendoza, M. Veloso, and R. van de Molengraft, "Skills, Tactics and Plays for distributed multi-robot control in adversarial environments," in *RoboCup 2017: Robot World Cup XXI*, (Cham), pp. 277–289, Springer International Publishing, 2018.

[8] B. Browning, J. Bruce, M. Bowling, and M. Veloso, "Stp: Skills, tactics, and plays for multi-robot control in adversarial environments," *Proceedings of the Institution of Mechanical Engineers, Part I: Journal of Systems and Control Engineering*, vol. 219, no. 1, pp. 33–52, 2005.

[9] R. T. Marler and J. S. Arora, "Survey of multi-objective optimization methods for engineering," *Structural and multidisciplinary optimization*, vol. 26, no. 6, pp. 369–395, 2004.

[10] J. K. Verma and V. Ranga, "Multi-robot coordination analysis, taxonomy, challenges and future scope," *Journal of Intelligent Robotic Systems*, vol. 102, no. 10, 2021.

[11] B. P. Gerkey and M. J. Matarić, "A formal analysis and taxonomy of task allocation in multi-robot systems," *The International journal of robotics research*, vol. 23, no. 9, pp. 939–954, 2004.

[12] Z. Yan, N. Jouandeau, and A. A. Cherif, "A survey and analysis of multi-robot coordination," *International Journal of Advanced Robotic Systems*, vol. 10, no. 12, p. 399, 2013.

[13] F. Duchoň, A. Babinec, M. Kajan, P. Beňo, M. Florek, T. Fico, and L. Jurišica, "Path planning with modified a star algorithm for a mobile robot," *Procedia Engineering*, vol. 96, pp. 59–69, 2014.

[14] I. Kostavelis and A. Gasteratos, "Semantic mapping for mobile robotics tasks: A survey," *Robotics and Autonomous Systems*, vol. 66, pp. 86–103, 2015.

[15] B. Kuipers, "The spatial semantic hierarchy," *Artificial Intelligence*, vol. 119, no. 1, pp. 191–233, 2000.

[16] A. Nüchter and J. Hertzberg, "Towards semantic maps for mobile robots," *Robotics and Autonomous Systems*, vol. 56, no. 11, pp. 915–926, 2008.

[17] M. Beetz, D. Beßler, A. Haidu, M. Pomarlan, A. K. Bozcuoğlu, and G. Bartels, "Know rob 2.0—a 2nd generation knowledge processing framework for cognition-enabled robotic agents," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 512–519, IEEE, 2018.

[18] T. J. Jech, T. Jech, T. J. Jech, G. B. Mathematician, T. J. Jech, and G.-B. Mathématicien, *Set theory*, vol. 14. Springer, 2003.

[19] P. van Lith, "Critiquing msl team behavior," 2021. unpublished.

[20] F. Goes, E. Schwarz, M. Elferink-Gemser, K. Lemmink, and M. Brink, "A risk-reward assessment of passing decisions: comparison between positional roles using tracking data from professional men's soccer," *Science and Medicine in Football*, pp. 1–9, 2021.

[21] J. Gudmundsson and M. Horton, "Spatio-temporal analysis of team sports–a survey," *arXiv:1602.06994*, 2016.

A Mu-field objectives

Table 3 gives the mu-objectives currently in use. In some cases they only apply to a certain role or skill, this is given in parentheses.

Table 3: The objectives in keywords used in the 3 types of mu-field

Parameter	Attacker advanced	Walk free	Defensive
1	Not in opponent goal area	Not in opponent goal area	not in goal areas
2	Not on opponents, based on opponent velocity	Not on opponents, based on opponent velocity	Not within a radius from the ball
3	Not too close to opponents	Not between the ball and goal	Not too close to peers
4	Don't drive through opponents	Not too close to peers, minimum pass distance	Not on opponents
5	Not on sidelines	Not on sidelines	On half circle around opponent
6	Free line to goal (skill: shot)	Free line to ball	On line between defensepoint and opponent
7	Drive towards aim target, the goal	Free line to goal (role: offensive)	Not on line between defensepoint and opponent
8	Distance to opponent goal (skill: shot)	Free space, keep distance from peers	Near Own goal
9	Drive to point of interest	Close to opponent goal (role: offensive)	Near opponent goal
10	Free line to peers (skill: pass)	In kicking region (role: offensive)	At own target
11	Not towards opponents (skill: human dribble)	Make field wide	Not in penalty area if a someone more important than you is already there
12	Stay away from the center of the field (skill: human dribble)	Position for directly giving a new pass to the next player	Close to the penalty spot
13	Stay away from the field borders (skill: human dribble)	Don't drive through opponents	Not in the 3m circle around the ball but close to it
14	Driving cost	Not between ball and potential pass receiver	Penalty area positioning (role: first defender)

15	Stay in area around center line (role: defensive)	On line between own goal and opponents on our half that do not have the ball
16	Driving cost	On line between ball and home goal
17	Stay way from the ball (skill: depth pass)	
18	Not behind opponents (skill: depth pass)	
19	Not behind peers (skill: depth pass)	
20	Not to close to the field lines, dependent on angle between ball and pass receiver	

B Risk-reward objectives

The formulas for calculating the different objectives used in the decision algorithm 1 are given here.

Opponents to goal

For determining the opponents between the position of the skill's target, or current position in case of a shot, the convex hull of this position and the opposing goal is taken. The amount of opponents intersecting with this regions is a measure for the opponents blocking the path to the goal.

Reward for scoring position

Equation 12 shows the equation used for computing the quality of the position. Here, the distance between the current position and the penalty spot $d(pos, penaltySpot)$ (deemed the best scoring position) is divided by half of the $fieldLength$. The closer to the penalty spot, the better the position is and this decreases linearly with the distance.

$$scoreReward = \begin{cases} (1 - \frac{d(pos,penaltySpot)}{(0.5 * fieldLength)}) * 100, & \text{for } d \leq (0.5 * fieldLength) \\ 0, & \text{otherwise} \end{cases} \tag{12}$$

Risk from pressure on the ball

Equation 13 shows the equation used for computing the risk of nearby opponents intercepting the ball. Here, the the area A of the composed feasible space divided

by a maximal area A_{max}.

$$pressureRisk = \begin{cases} (1 - \frac{A}{A_{max}}) * 100, & \text{for } A \leq A_{max} \\ 0, & \text{otherwise} \end{cases} \tag{13}$$

Risk of the length of the skill

Equation 14 shows the equation used for computing the risk for the length of an skill. The longer a skill takes, the more time opponents have to intercept the ball or reorganize themselves. The distance between the current position and the skill target $d(pos, target)$ (deemed the best scoring position) is divided by the $fieldLength$. The risk of a skill increases linearly with the distance to its target.

$$lengthRisk = \begin{cases} (1 - \frac{d(pos,target)}{fieldLength}) * 100, & \text{for } d \leq fieldLength \\ 0, & \text{otherwise} \end{cases} \tag{14}$$

Weighted sum of the risk objectives

The total risk is applied as a normalized weighted sum of the risk objectives, varying between 0 and 100. The weight factors w_1 and W_2 are used to account for the importance of the objective. The formulation used is depicted in equation 15

$$totalRisk = \frac{w_1 * pressureRisk + w_2 * lengthRisk}{(w_1 + w_2)} \tag{15}$$

C Convex hull and Voronoi diagram

The concepts of the convex hull and Voronoi diagram are explained here. Both concepts are implemented by applying functions from the used Shapely package.

Convex hull

The convex hull is a geometrical concept, and may be defined as the intersection of all convex sets containing a given subset of a euclidean space. For a bounded subset of the plane, as in this work, it may be visualized as the shape enclosed by a rubber band stretched around the subset.

Voronoi diagram

The Voronoi diagram can be described as the partitioning of a plane with n points (generating points) into convex polygons such that each polygon contains exactly one generating point, and every point in a given polygon is closer to its generating point than to any other.

D Experimental settings

The settings used performing simulation of which the results are shown in this work are given in table 4, as can be seen below.

Table 4: Settings used in the performed tests

Features	
Diameter Turtle	$0.5[m]$
Diameter opponent	$0.5[m]$
Avoid diameter Turtle	$0.8[m]$
Avoid diameter opponent	$1.2[m]$
Field dimension	$22.401 \times 13.890[m]$
Goal width	$2.402[m]$
Penalty spot	[0, 7.640] $[m]$
Skill regions	
f_v	0.5
v_p	$5[m/s]$
v_t	$4[m/s]$
r_{min}	$2[m]$
a_{max}	$4.5[m/s^2]$
r_t	$0.05[s]$
Minimal pass distance	$3[m]$
Dribble region radius	$3[m]$
Tactics	
w_1	0.8
w_2	0.4
A_{max}	$3[m^2]$
Decision making	
shot reward threshold	70
pass reward threshold	70
dribble reward threshold	75
pass risk threshold	50
dribble risk threshold	30