
# Evaluation and Enhancement of Artificial Potential Fields for Path Planning in Dynamic Environments

Bachelor's thesis
in partial fulfilment of the requirements for the degree of
Bachelor of Science

University of Münster
Computer Science Department

Supervisor:
*Prof. Dr. Malte Schilling*

First assessor:
*Prof. Dr. Malte Schilling*

Second assessor:
*Prof. Dr. Paula Herber*

Submitted by:
*Niklas Hellmann*

Münster, October 2023

## Abstract

Driving through an area without collision is not only a very important topic in our daily lives but also in robotics. The artificial potential field is a well-known path planning model for static environments. This work examines the extent to which this model can be transferred to an environment with moving obstacles. Since this does not produce good results, three extensions are used, which, among other things, use pre-projections and relative speeds for the calculation. These models are first tested in a purely theoretical environment with many simplifications in order to gain parameters for them and are subsequently tested in a 3D simulation called Gazebo. It turns out that all extensions reduce the number of collisions but by far the model with relative velocities works best.

# Contents

# 1 Introduction

Nowadays, more and more machines are expected to be able to work autonomously, i.e., without additional human influence. Be it in the virtual world to fend off cyber attacks and answer questions of any kind or in the real world to transport people and to explore foreign planets. Autonomous driving, which is already possible in some vehicles, must of course solve many technical problems as well as ethical issues: the detection of obstacles such as pedestrians or other cars, the calculation and implementing goal-oriented evasive maneuvers, or generally approaching a specific destination. These are only a few of such problems but they quickly show how complex a seemingly simple task like "driving from A to B through an unknown area" can be.

Path planning is therefore about maneuvering the robot safely to the target without causing collisions or predicting and preventing them at an early stage. In an unknown environment, the robot must constantly update its route and adapt to the new conditions detected by its sensors. In this case, it usually cannot be assumed that the robot runs along an optimal path as initially pre-planned, in particular when there is intense traffic. With previous knowledge of the initial situation, however, this can be achieved. Depending on how well the environment and its changes over time are known, a plan can be calculated in advance, which basically only needs to be executed. A plan can also be calculated initially, which may turn out to be impractical due to changes in the environment that could not be calculated beforehand, making it necessary to constantly re-plan. This division is based on the given environmental information, whether only a small local part is known or the whole map. It is called local and global path planning. Local calculations are, however, for real time applications rather suitable, since it is with applications rarely the case that the entire environment is known and it is likewise difficult to be able to react to unexpected events.

In the following work, a popular model in path planning called "artificial potential fields" (APF) will be presented and analysed. The environment detected by the robot is converted into a vector field which is permanently updated when the robot moves. It stands out from other models because of its simplicity and robustness and can also be used for real time applications. This model delivers good results especially in a static environment, i.e., an environment where the obstacles do not move. However, the classic model is not designed for moving obstacles, which is why it performs rather poorly in dynamic environments. This issue will be investigated, adding several possible extensions to the APF and analyzing its usefulness.

## 1.1 Related Work

Path planning is a very complex topic with a wide variety of approaches, so the different types are only roughly presented in this paper. These can be divided into two types: model-based and learning-based. In the former, a search space for the planning is built, for example a two- or three-dimensional one to describe the environment. In the case of a dynamic environment, a four-dimensional one can be used, where time is added as an additional dimension. Now a path through this environment must be found by means of mathematical analysis [1]–[3]. For example, a grid-based environment map can use various well-known search algorithms such as the Dijkstra algorithm [4] or wavefront expansion algorithm [5].

Another approach that has become increasingly popular in recent years is the use of machine learning. Mostly, neural networks are trained using raw sensor data, such as camera images or LIDAR sensors. Supervised learning-based approaches or reinforcement learning-based approaches are just some of the examples. But they are all based on a high amount of data, which is needed to learn. With enough different data sets, the method can abstract well enough and work reliably even in previously unknown areas. It can operate in static as well as in dynamic environments [6].

## 1.2 Thesis outline

The following work is structured as follows: First, a brief explanation of the classic model of the APF is provided and problems that should be considered. In chapter 3, the testing methodology is presented and the different extension models to be tested are introduced. This is followed by tests on these with a purely theoretical approach in chapter 4 with many simplifications of the environment. In chapter 5, the models are then tested using Gazebo and ROS with the Turtlebot3, which should provide a more realistic evaluation of the models. Chapter 6 deals with the evaluation of the previous tests and compares the models. The last chapter contains the final conclusion and gives an outlook on further improvements.

# 2 Background

Since the Potential Field algorithm is the basis of this work and all extensions are based on it, it will be explained in the following section. In addition, the transfer of the algorithm results into a robot movement is also briefly addressed.

## 2.1 Potential Field

The Artificial Potential Field (APF) was firstly introduced by O. Khatib in 1985 [7]. He described a method for path planning in which the robot is under the influences of two different types of constructed fields generated by the destination point and obstacles nearby: one to move the robot to the target and the other to keep it away from hindrances. This approach for navigating a robot through an environment later became important for path planning, not only because of its ease of implementation, but also because of its good mathematical analysis capabilities. Its simplicity also allows it to work in real time.

You can use different types of forces here. However, the most common ones are presented below. The first constructed field is determined by the target point and the position of the robot. The field is created so that the robot can 'roll down' the field similar to a marble placed on a hill and rolling down its slope and thus reach the target. This becomes clearer by having a closer look at figures 2.1a, 2.3a and 2.3b below. The following definitions in this section are based on [8].

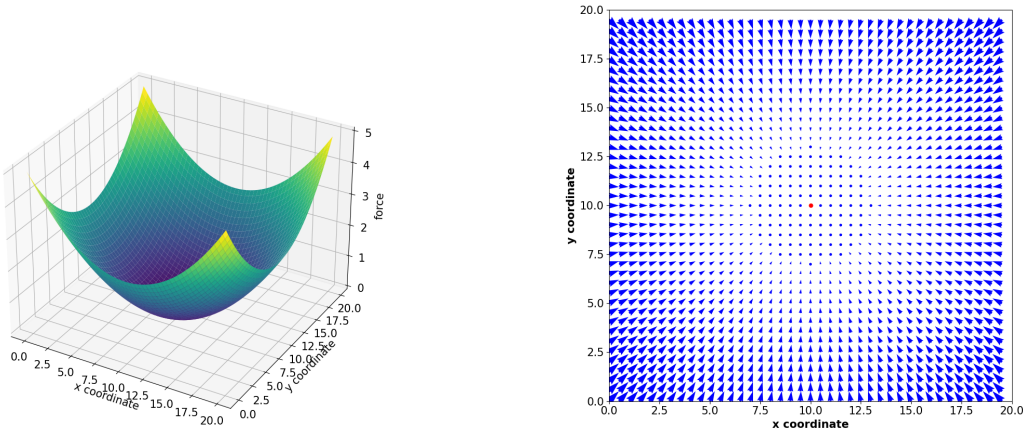
**Definition 2.1.1** (Attraction Field)
Let $p \in R^n$ be a point in the environment representing the position of the robot on a two dimensional map. Let $g \in R^n$ be the goal of the path planning. The attraction field at p can be represented as:

$$U_{att}(p) = \frac{1}{2} \cdot k_a \cdot ||p - g||^2$$

where $k_a$ is a positive scaling constant and $||.||$ is the Euclidean distance.

The force acting through the field is calculated by taking the negative gradient.

$$\begin{aligned} F_{att}(p) &= -\nabla U_{att}(p) \\ &= -\frac{1}{2} k_a \cdot 2 \cdot (p - g) \\ &= k_a \cdot (g - p) \end{aligned}$$




(a) Attraction field

(b) Vector field of attractive forces. Goal point in red in the middle.

Figure 2.1: Visualizations of the attractive forces on the same map. a) shows the attraction field and b) the resulting forces.

Another field is generated by the obstacles of the environment and generates repulsive forces from them. Here, too, different variants can be used, but the most common one is based on the gravitational force.

**Definition 2.1.2** (Repulsive Field)
Let $p \in R^n$ be a point in the environment representing the position of the robot on a two dimensional map. Let O$\subset R^n$ be a set representing the position of k obstacles.
For each obstacle position $o_i \in O$ (i = 1,...,k) the repulsive Field is calculated as:

$$U_i(p) = \begin{cases} \frac{1}{2} \cdot k_r \cdot (\frac{1}{d} - \frac{1}{d_0})^2 & , d \leq d_0 \\ 0 & , d \geq d_0 \end{cases}$$

where $k_r$ is a positive scaling constant, $d_0$ the distance of the obstacle repulsive force field and d being the Euclidean distance between $o_i$ and the robot position p.

The force acting through each obstacle is calculated by taking the negative gradient.

$$F_i(p) = \begin{cases} -\nabla U_i(p) & , d_0 \leq 0 \\ 0 & , d_0 \geq 0 \end{cases}$$
$$= \begin{cases} k_r \cdot (\frac{1}{d} - \frac{1}{d_0}) \cdot (p - o_i) \cdot \frac{1}{d^3} & , d \leq d_0 \\ 0 & , d \geq d_0 \end{cases}$$

The total repulsive field is the combination of each obstacle field:

$$U_{rep}(p) = \sum_{i=0}^{k} U_i$$

This results in the total repulsive force acting on the robot as:

$$F_{rep}(p) = \sum_{i=0}^{k} F_i$$

The combination of the two fields now results in the potential field of the environment as shown in figure 2.3 below.

**Definition 2.1.3** (Potential Field)
Let $p \in R^n$ be a point in the environment representing the position of the robot on a two dimensional map. Let O$\subset R^n$ be a set representing the position of k obstacles. Let $g \in R^n$ be the goal of the path planning.

The APF function can be represented as:

$$U(p) = U_{att}(p) + U_{rep}(p)$$

and this gives the forces acting on the robot:

$$F(p) = -\nabla U(p)$$
$$= F_{att}(p) + F_{rep}(p)$$

Having previous knowledge about the whole environment makes it possible to generate the potential field offline and to further decrease the computation.

If this is not the case, the positions of the obstacles as well as those of the robot must be constantly updated. Thus, there are constantly new forces acting on the robot, which ultimately determine its path.

### 2.1.1 Limitations

Nonetheless, this classic model also has its flaws. The first one being that the goal is not reachable when an obstacle is too narrow resulting in the repulsive force being significantly higher than the attraction.

However, the most common problem is the so called local minimum, also resulting in a not reachable goal situation. It can occur, for example, when a passage created by two or more obstacles is too narrow, not specifically for the physical robot itself, rather the fact that the forces resulting from the hindrances are once more significantly higher than the attraction by the goal. The robot is now trapped in a loop right in front of these obstacles or does not move at all.

Figure 2.4 illustrates these two problems. How these problems can be solved and others can be found in [9].

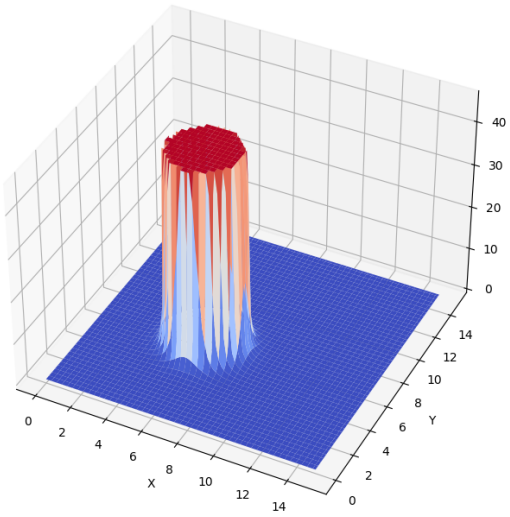
## 2.2 Robot Controls

The method described so far provides at the end a vector in which the robot should move. However, this vector must first be processed further, since it is not generally applicable in this form. Depending on the robot constraints, this vector must be adapted to the conditions of the vehicle. For example, a car has only 2 degrees of freedom, the linear speed and steering angle, although it moves in a 3-dimensional space. Even if one would break this down to two dimensions, a classic car could not simply drive in any desired direction. This is called a nonholonomic vehicle. Thus, the resulting motions of the robot are at best approximations of the desired vector and can be determined by means of geometric relations.

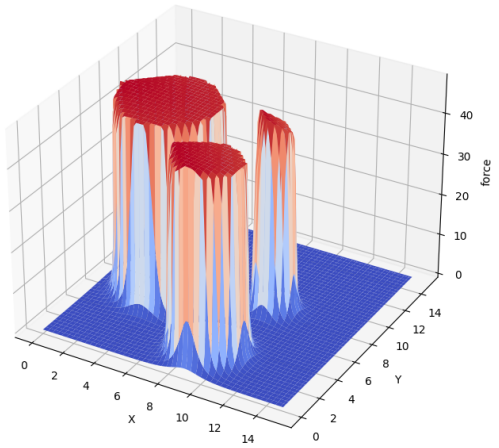
The Turtlebot3 robot, which will later be used for simulations, is such a nonholonomic vehicle. In order to control it, it needs the linear speed $v$ it is supposed to travel and the rotation $\omega$ it is meant to make. If it is now known in which direction $(x, y)^T$ the robot should move, the control inputs can be calculated taking into account the current orientation $\phi$ of the robot to the movement point:

$$\begin{pmatrix} v \\ \omega \end{pmatrix} = \begin{pmatrix} cos(\phi) & sin(\phi) \\ -\frac{1}{b}sin(\phi) & \frac{1}{b}cos(\phi) \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix}$$

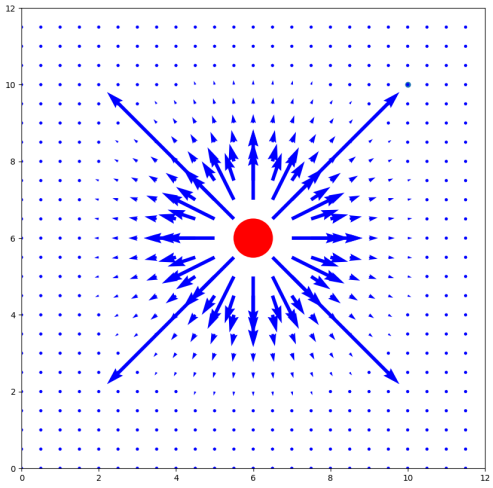
Here $b$ can be taken as the length of the direction vector. The rotation speed can be controlled by $b$. This calculation is based on [10]. For a better understanding of this topic and the mathematical calculations behind it, please refer to [10] and [11].




(a) Repulsive field

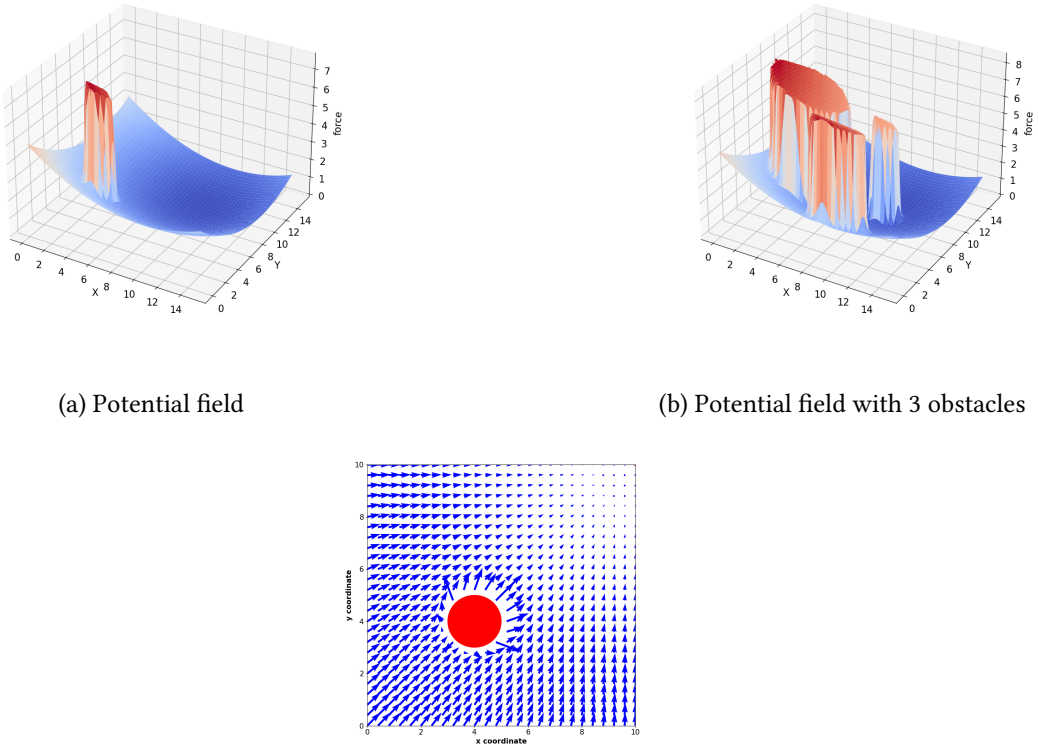



(b) Repulsive field with 3 obstacles




(c) Force acting on the robot with one obstacle

Figure 2.2: Visualizations of the repulsive forces. a) and b) show the field with different numbers of obstacles, c) the resulting forces from a).

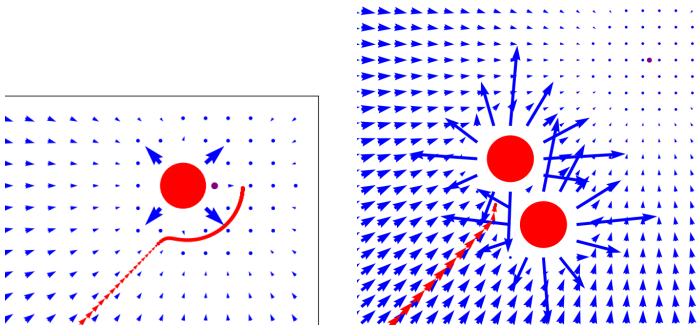



(a) Potential field

(b) Potential field with 3 obstacles

(c) Force acting on the robot with one obstacle

Figure 2.3: Visualizations of the total potential field consisting of attractive and repulsive forces. a) and b) show the field and c) the forces resulting from a).



(a) Example of unreachable goal

(b) Example of locale minima

Figure 2.4: Limitation examples. Purple dot shows the destination point. Red circles are obstacles. On the left, the target cannot be reached because the obstacle has too much influence on the area near the target. On the right, the destination is not reached because the route has led to a local minimum.

# 3 Method

In the following, the general experimental setup is explained first, followed by the presentation of various extensions, which will be tested afterwards.

## 3.1 Settings

In order to test the APF algorithm and various extensions with respect to a dynamic environment, the following experimental setup is used: A robot starts on a map with sufficient distance to its target. Between it and the target is a square area also sufficiently large with several cylindrical obstacles, all of which can move at the same and constant speed. When the cylinders reach the boundaries of the area, they simply bounce off them without losing speed.

This setup makes it possible to test the different models at varying speeds and with varying numbers of obstacles to determine their limits. A cylinder is well suited as an obstacle because it has a very simple shape and can be used relatively easily for distance calculations or sensor recognition.

The tests are divided into different sections. First, a purely theoretical simulation is tested, where many simplifications of the environment and the robot are assumed. After that, a robot test environment called Gazebo is used, which then also considers a real robot model.

The robot used will be a Turtlebot3 Burger since this is a frequently used model in research and already has many software modules, which makes it easier to work with. The Turtlebot3 is a differential-drive robot with a nonholonomic constraint on its velocity, which means that it cannot move freely in any direction, similar to cars.

## 3.2 Models

Now the different variants of the potential field algorithm are presented. The resulting force of the potential field can be described by the equation

$$F(E) = F_{att}(E) + F_{rep}(E)$$

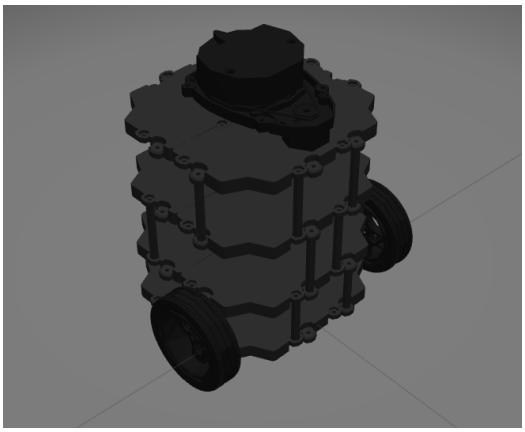


Figure 3.1: Image of a Turtlebot3 in the Gazebo robot environment

for all variants, where the environmental variables $E$ as a set of variables can differ in number and intensity. Thus, they all have an attracting and a repelling part. However they take into account the environment in different ways so that different routes through the obstacle course are created. These environment variables $E$ have to be changed at each calculation step in order to adjust the movement of the robot as best as possible to the environment.

### 3.2.1 Artificial Potential Field

In this classic variation of APF, an attractive field is generated towards the target (definition 2.1.1) and repulsive fields are generated away from obstacles (definition 2.1.2) Together, they produce a vector field on the environment that defines the motion of the robot (definition 2.1.3). The movement of the obstacles is thus not taken into account. Hence, a new repulsive field results with each calculation step since the positions of the objects changes.

### 3.2.2 Forward Projection APF

In this extension, the attractive forces as in definition 2.1.1 remain the same as in the classic model, only the repulsive ones change.

In the forward projection, the speed of the obstacles is now added to the calculation of the repulsive field. It is called forward projection because artificial obstacles are added in addition to the actual obstacle. These additional objects are located in the direction of movement of the obstacle and should prevent the robot from getting in the way of the obstacle.

The first thought that comes to mind is to place a series of artificial objects of the same size in front of the real object and treat them like real obstacles, each with a distance of one movement step per time unit.

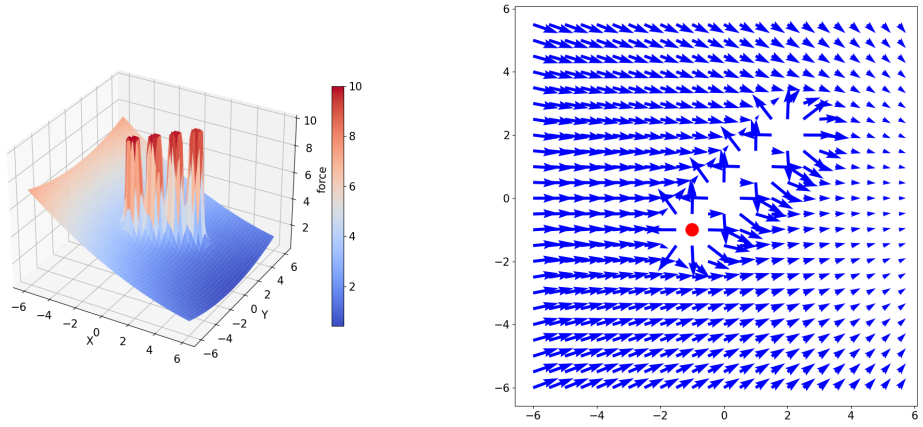



Figure 3.2: Potential field and vector field of the forward APF with an obstacle at (-1,-1) which moves with the vector (1,1) if only three more obstacles are artificially added to take the movement into account.

However, this raises several problems. On the one hand, these projections create huge velocity vectors for the robot in places where this would not be necessary, because there are only theoretical objects. On the other hand it could result in a local minimum for a calculation point which is exactly between two projections, whereby the robot does not move at all and thus does nothing to prevent the collision. Moreover, the motion vectors could result exactly in the direction of the actual obstacle, which would also be unsuitable.

For the repulsive forces, basically only positions and the distances to these obstacles are relevant. So, to solve the problem of these gaps in the fields and to allow a more continuous projection, one can simply take the orthogonal distance to the line of motion of the obstacle for a desired point. For the position of the artificial obstacle for the calculation as in definition 2.1.2 one takes the nearest possible point on the movement line. This way it is possible to clear the space in which the object is moving more precisely, because the robot is always pushed away from the line of motion.

However, here it is not weighted where the obstacle is located. In order to give less weight to the obstacles farthest away from the actual obstacle, the reference point of the calculation is shifted further away depending on the distance to the actual object, thus, increasing the distance to it and decreasing the forces.

For later use, the parameter future_count is introduced which describes the number of the motion

vector for calculating the last projection. So, the calculation of the last projection looks like this:

$$position_{projected} = pos_{current} + future\_count \cdot movement_{current}$$

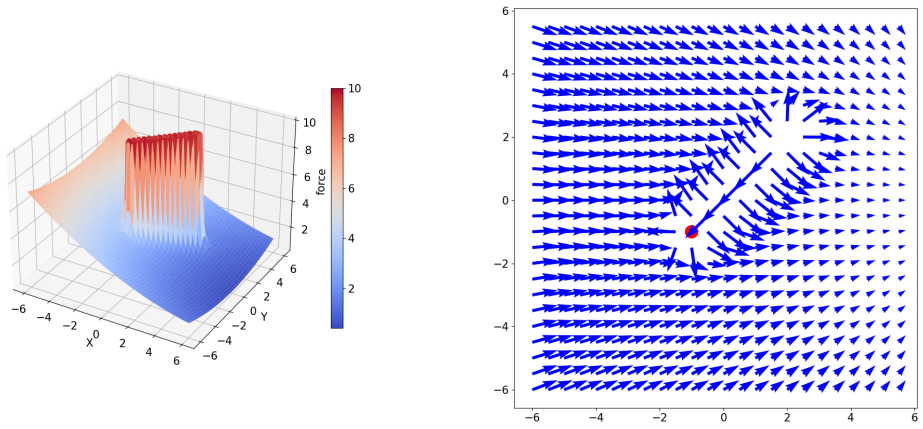



Figure 3.3: Potential field and vector field of the forward APF with an obstacle at (-1,-1) which moves with the vector (1,1) when a continuous repulsion calculation takes place.

### 3.2.3 Rotational Forward APF

This extension builds upon section 3.2.2 and only adds the rotation of the repulsive forces. Therefore, the calculation of the attractive force is the same as in definition 2.1.1.

The repulsive vectors are calculated as in section 3.2.2 but they receive a bit of rotation depending on the motion of the object. This idea is based on [12]. The angle $\alpha$ by which the vectors move must be defined beforehand. In general, it is assumed that this angle is positive. This is not mandatory, however, the following calculations would have to be adjusted accordingly. Since the vector must not be rotated too much, because otherwise it points in the direction of the object and only positive rotations are considered, $\alpha$ must lie between 0 and 90 degrees.

The rotation of the force vector is now calculated as follows:

$$F^{\star} = \begin{pmatrix} cos(\alpha) & -sin(\alpha) \\ sin(\alpha) & cos(\alpha) \end{pmatrix} \cdot F_{forward}$$

$\alpha \in [0, \pi/2]$ and $F_{forward}$ the calculated force vector as in section 3.2.2. Positive values for the angle provide a counterclockwise rotation, negative for a clockwise rotation.

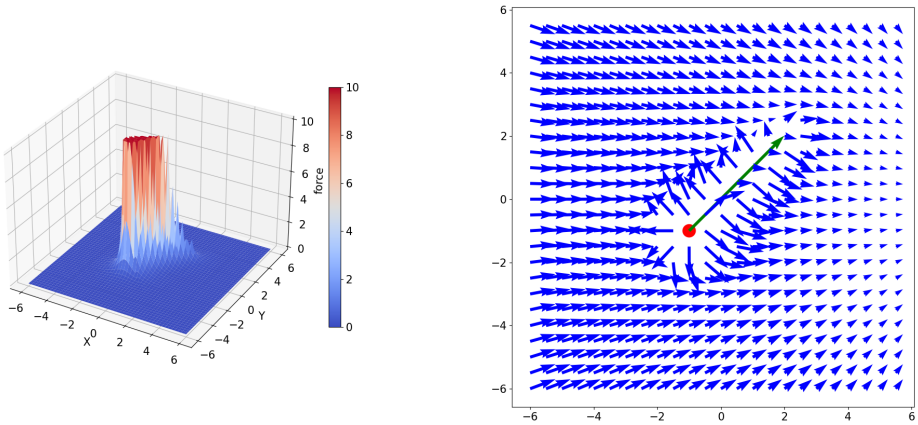



Figure 3.4: Potential field and vector field of the forward APF with an obstacle at (-1,-1) which moves with the vector (1,1) when the repulsive forces decrease at later projections.

Alpha is therefore adjusted again to the movement of the obstacle. Looking from the robot to the target, it is checked whether the obstacle is moving to the right or to the left. In the former case, alpha remains the same, otherwise it is negated. This is to create a rotation field around the object, where the robot is guided along behind the object.

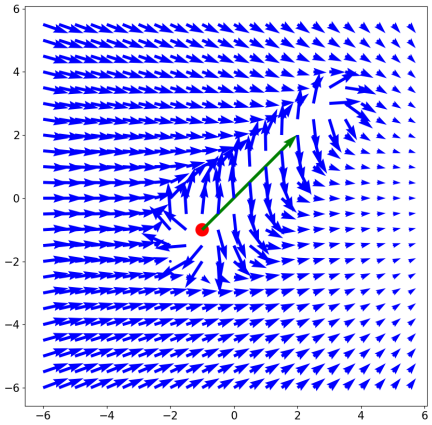


Figure 3.5: Potential field and vector field of the rotational APF with an obstacle at (-1,-1) which moves with the vector (1,1). The repulsive vectors are rotated by 40°.

### 3.2.4 Improved APF

The last method considered is from an elaboration on the application of the improved APF algorithm for underwater vehicles [13]. In this paper, several extensions are presented which, in addition to dealing with moving obstacles, also take into account local minima and strong repulsive forces near the target (see section 2.1.1). The latter aspects do not play a major role for this work and the presented experimental setup, since local minima dissipate quickly due to the moving obstacles and the target is not in the range of motion of the obstacles.

The attractive force remains the same as in the classic APF but the repulsive forces are again adjusted. Thus, in addition to the classic repulsive force in the definition 2.1.2, another term is added to account for the different motions. The new field of repulsive forces is defined as follows:

$$U_{new}(p) = \begin{cases} U_i(p) + U(V) & , d \leq d_0 \ \ \& \ \ v_{ao} \geq 0 \\ 0 & , else \end{cases}$$

where

$$U(V) = k_v \cdot \frac{(v - v_o)^T e_{ao}}{d} = k_v \cdot \frac{v_{ao}}{d}$$

$d$ is the distance between the robot and the obstacle. $e_{ao}$ is the normalized vector between the position of the robot and the obstacle. $v$ the motion vector of the robot and $v_o$ that of the obstacle. $k_v$ is a scaling term to be able to adjust its effect. $U_i(p)$ as in definition 2.1.2.

$v_{ao}$ is therefore the relative velocity component in the direction of the obstacle. If this is greater than 0, the objects move towards each other. If this is not the case, no repulsive force has to be considered since this is not necessary for a non-collision course.

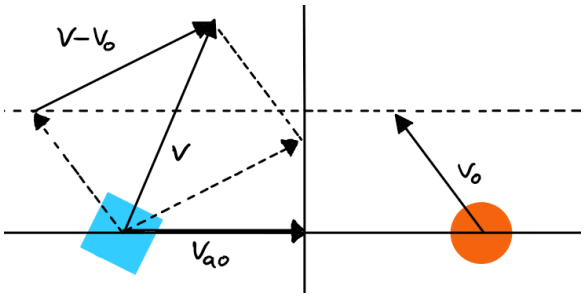



Figure 3.6: Trigonometric relationships of relative velocity and $v_{ao}$. Blue square on the left represents the robot and the orange circle represents the moving obstacle. $v$ is the velocity of the robot and $v_0$ of the obstacle. $v_{ao}$ is the component of the relative velocity pointing towards the obstacle. (Based on [13])

The repulsive forces result as follows:

$$
\begin{aligned}
F_{new}(p) &= -\nabla U_{new}(p) \\
&= \begin{cases} F_i(p) + F(V) & , d \leq d_0 \ \ \& \ \ v_{ao} \geq 0 \\ 0 & , else \end{cases}
\end{aligned}
$$

$$
F(V) = -k_v \cdot \frac{v_{ao} \cdot e_{ao}}{d}
$$

# 4 Theoretical Simulation

The next chapter is primarily intended to get a feeling for the different parameters of the models in order to work with them in the robot simulation in the following chapter.

## 4.1 Settings

First, the different models are tested on a purely theoretical environment, on a 2 dimensional map. So there is no use of a concrete robot. The movements are therefore only determined by the results of the different vectors of the models. However, they are throttled to a certain length of 1, if necessary. The size of the robot is set to 0.1. The size of the obstacles is twice the size of it. As described in the section 3.1 and sketched in figure 4.1, the robots start at the point (-10,0). The target is located at (10,0) and is considered to be reached, when a robot falls below the minimum distance of 1 to it. In this test, the obstacles move at varying speeds in a 6x6 square with the center at (0,0). Obstacles start on random positions within this field and move in any direction. Collisions with these obstacles do not affect the movement of the robots, which must be taken into account in the analysis. In addition to the distance traveled and the number of collisions, the time steps required by the different models for the same scenario are also analyzed.

A test run with set parameters for a selected model consists of 20 test configurations, each of which is tested with 100 runs. The configurations result from the combinations of the number of obstacles, which range from 2 to 6, and the obstacle speed, which can take the values 0.5, 1, 1.5 and 2.

## 4.2 Models

The test results of the different models follow, with each model being tested on a wide range of parameters in order to find satisfactory parameter pairs.

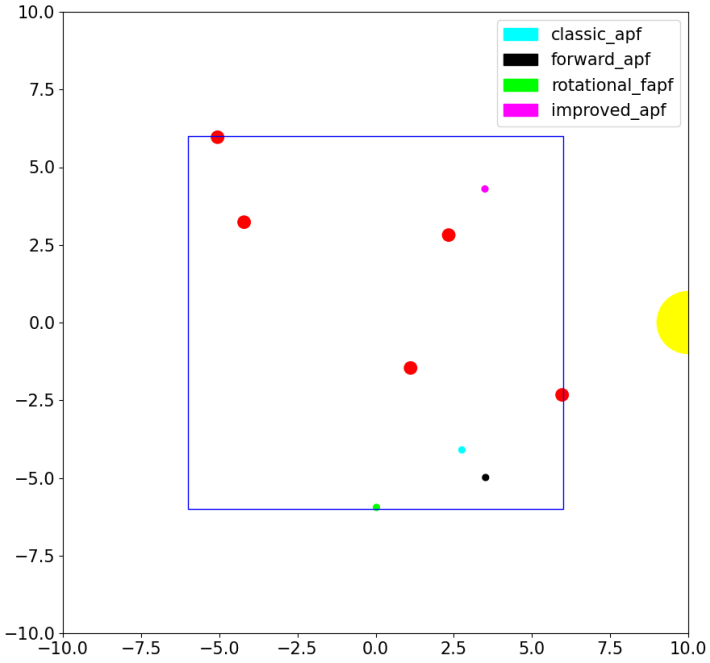



Figure 4.1: Sketched map with red circles as obstacles and yellow semicircle as target. The other different colored dots show different robot models. The rectangle of movement of the obstacles is shown in blue

### 4.2.1 Artificial Potential Field

First, the classic model is considered with $k_a = 1/20$. This parameter has proven to be good for the selected card size and will not change for later models. The parameter $d_0$, which defines the maximum distance of the repulsive forces, is first initialized with $d_0 = 3$. The focus is put on the scaling parameter $k_r$ of the repulsive forces. This parameter will take the values 0.5, 1, 3, 5, 10 and 20 during the first test runs.

During the test runs, it was observed that the average number of collisions calculated over the entire test run is reduced with higher $k_r$ as shown in figure 4.2. However, the number of average collisions does not seem to increase significantly even by further increasing the parameter beyond a certain point.

Nonetheless, since a test run covers many environmental parameters, it should additionally be divided into the different speeds of the obstacles here.

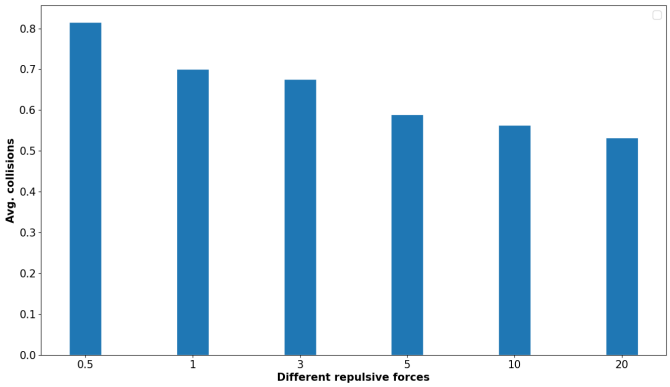



Figure 4.2: Average collisions with different repulsive forces as parameter on the x-axis when the classic APF was used.

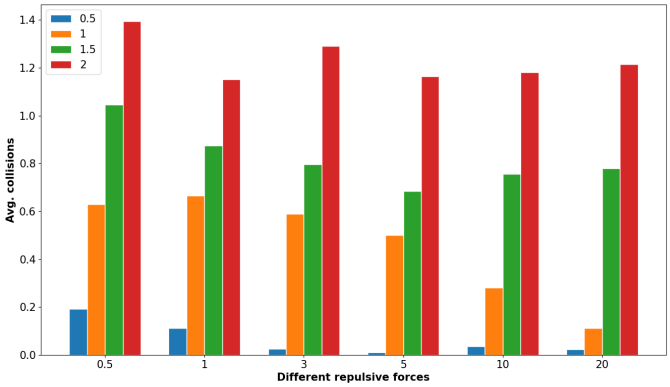


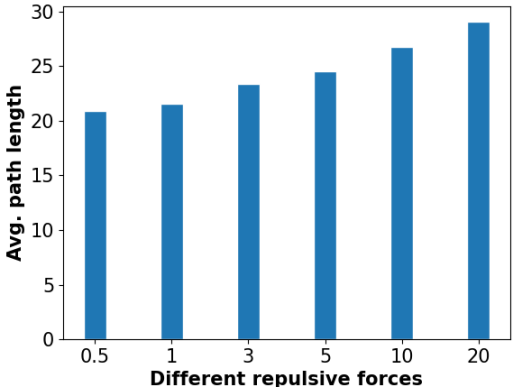

Figure 4.3: Average collisions with different repulsion forces as parameters on the x-axis and also separately for different obstacle speeds when the classic APF was used.

In figure 4.3 we can now see that the collision reduction seen earlier is not distributed over all speeds. Increasing the parameter even seems to favor the collisions. The results at $k_r = 5$ are always better than at $k_r = 20$ except for one speed, i.e. $v = 1$.

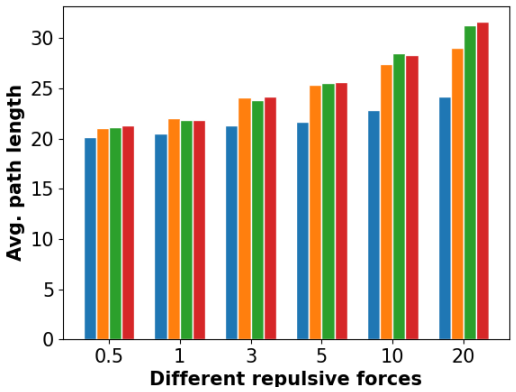
If we now look at the calculation steps as well as the path lengths, it is noticeable that these increase constantly and this also happens equally at the different speeds, cf. figure 4.4.

Next, $k_r = 5$ is assumed to be fixed and the effects of the parameter $d_0$ are analyzed. $d_0$ will take the values 1,3,5 and 10.

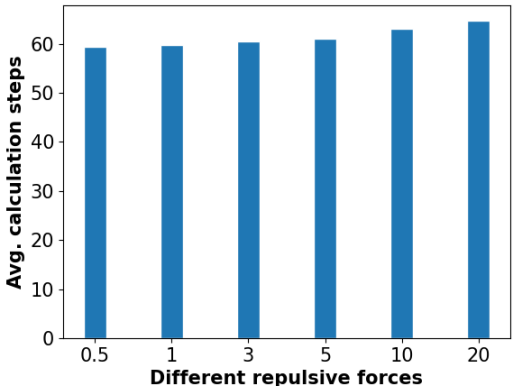
As can be seen in figure 4.5, the number of collisions decreases while increasing the parameter. However, this increases the number of calculation steps as well as the average path length. This



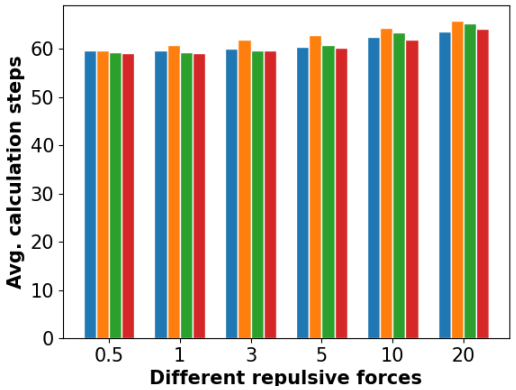

(a) Average path lengths for each repulsive force




(b) Average path lengths per obstacle speeds




(c) Average calculation steps for each repulsive force




(d) Average calculation steps for each repulsive force separated in different obstacle speeds

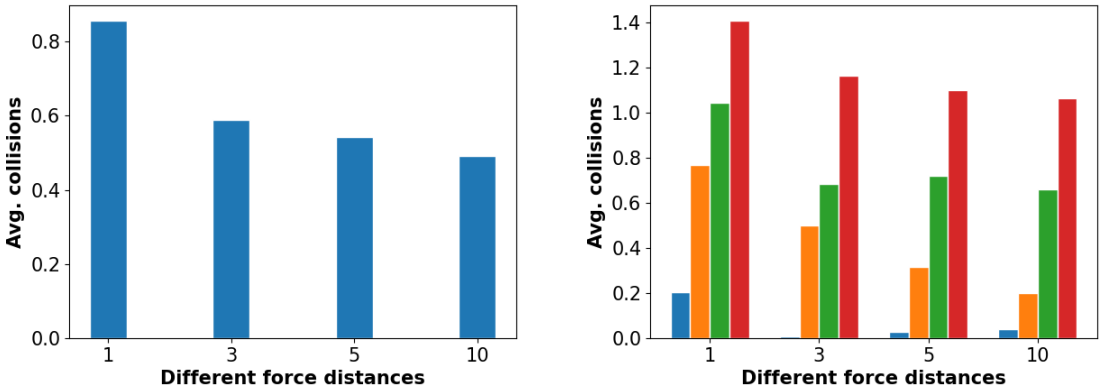
Figure 4.4: Average Path lengths and calculation steps of the classic APF. In b) and d) the different colors represent the different obstacle speeds (blue=0.5, orange=1, green=1.5, red=2).

tendency can also be seen in the division into the different obstacle speeds.

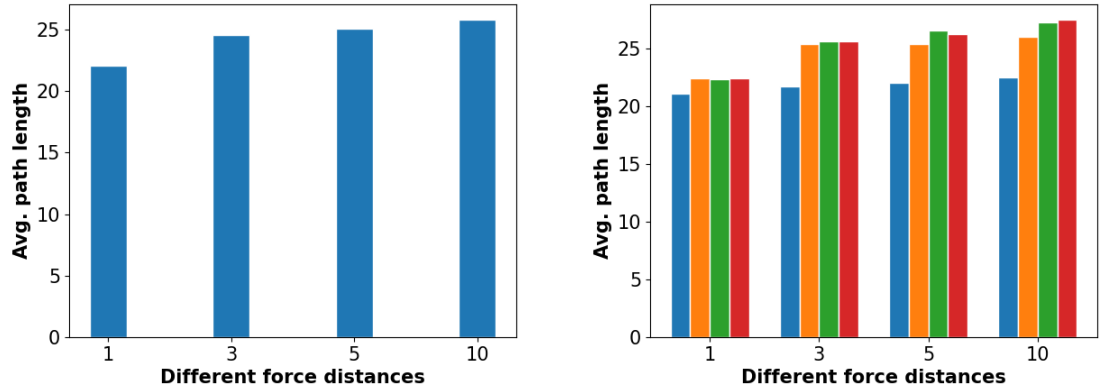
### 4.2.2 Forward Projection

Next, the Forward Projection APF algorithm is tested. For this, $k_r = 5$ and $d_0 = 3$ are assumed. It is to be tested to what extent the future count in the calculation affects the path. The values 0 to 5 were tested.

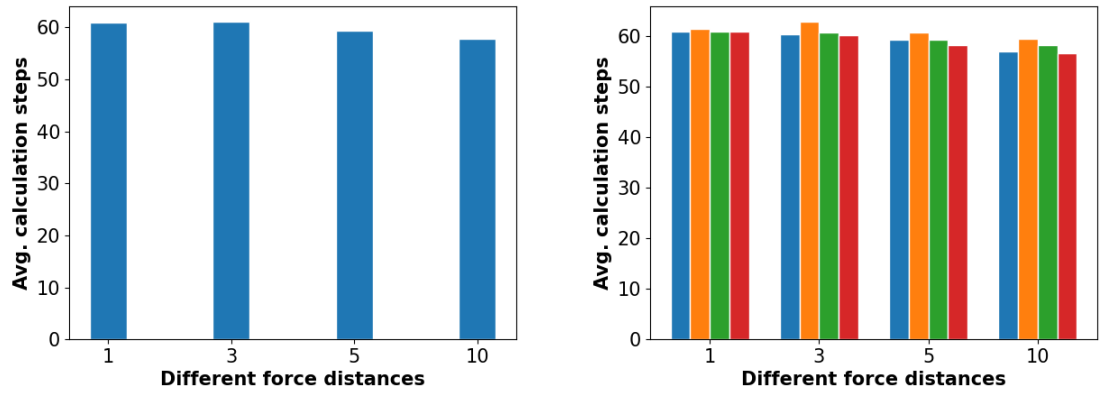
It can be seen that projecting the velocity even only with $future\_count = 1$ significantly reduces the number of collisions. Further increasing of the parameter causes less collisions in the beginning, but stagnates approximately from $future\_count = 3$, cf. figure 4.6a. When looking at the individual speeds, it also becomes apparent that higher future counts significantly reduce collisions, so that




(a) Average collisions

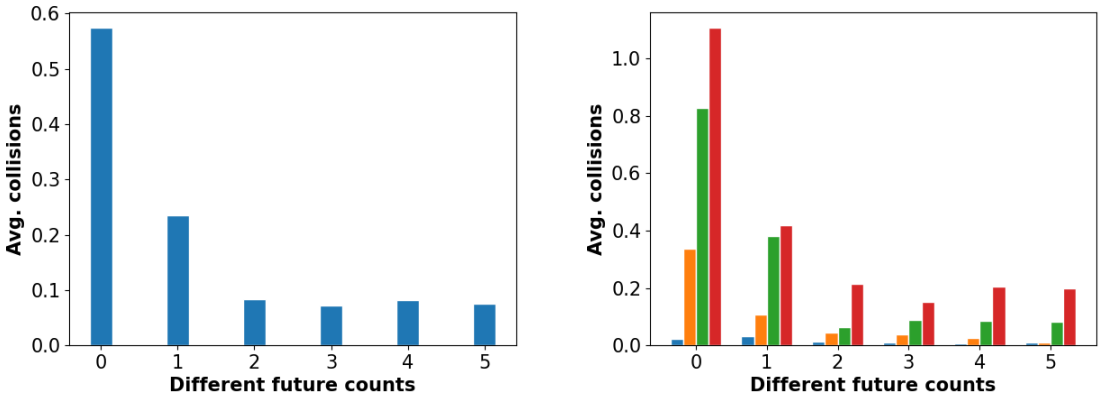



(b) Average path lengths

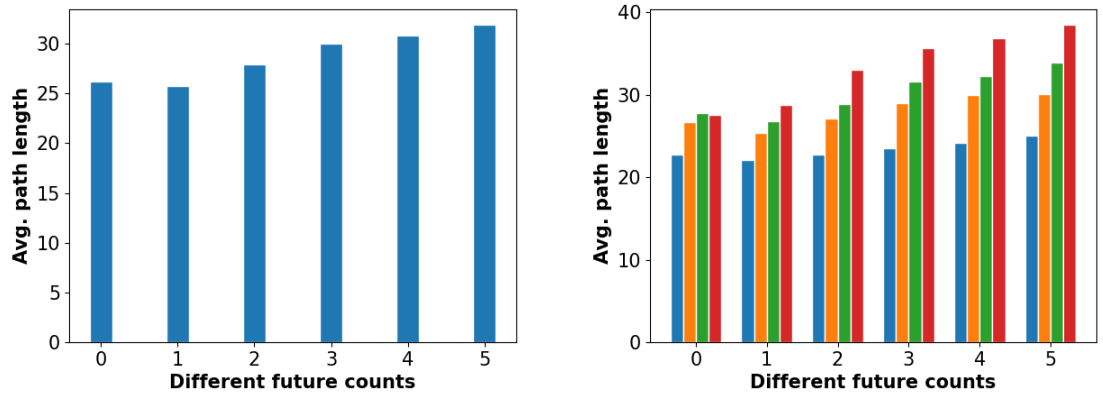



(c) Average calculation steps

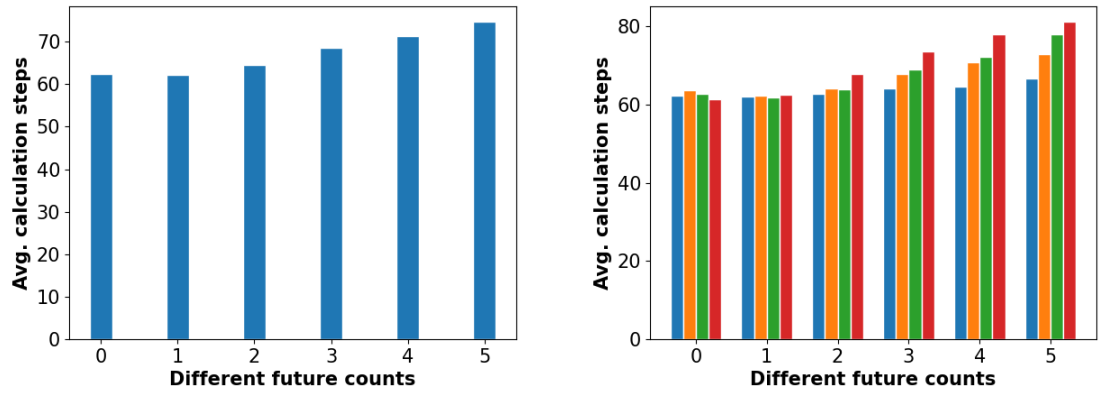
Figure 4.5: Results for the classic APF when changing the repulsive force distance. Left: Average results for the entire test run for different repulsive force distances Right: Average results for the entire test run for different repulsive force distances and separated in the different obstacle speeds. Different colors represent the different speeds (i.e. blue=0.5, orange=1, green=1.5, red=2).




(a) Average collisions




(b) Average path lengths




(c) Average calculation steps

Figure 4.6: Results for the forward APF when changing the future count of the projections. Left: Average results for the entire test run for different future counts. Right: Average results for the entire test run for different future counts and separated in the different obstacle speeds. Different colors represent the different speeds (i.e. blue=0.5, orange=1, green=1.5, red=2).

with a value of 5, not a single collision was recorded at the two lowest speeds. Also, the two higher speeds are only a fraction of the values of no projection.

However, the number of calculation steps and the length of the completed path increase. Especially at the higher speeds, these values increase strongly compared to the lower speeds.

### 4.2.3 Rotational APF

The extension by rotation fields is now tested with $k_r = 5$, $d_0 = 3$ and $future\_count = 3$. The influence of different angles is checked.

When looking at the collisions as shown in figure 4.7a, one can notice that an angle of 0° works best. As the angle is increased, the number of collisions also increases consistently until 35° is reached. There, the collisions are the second lowest, a bit lower then 20°. Increasing the angle further causes the collisions to increase rapidly from there on. Thus, at the end of the tolerated angles at 90° (as mentioned in section 3.2.3), the values are 1.75 on average.

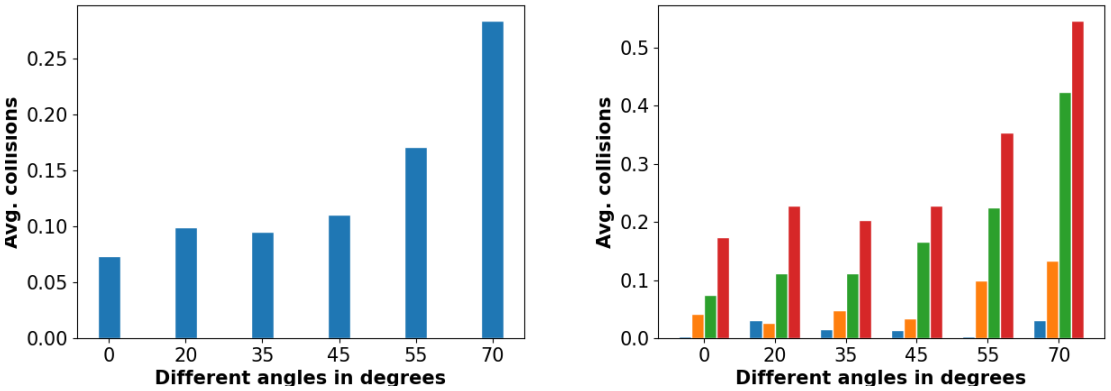
If we look again at the different obstacle speeds, we can see that it is not so easy to assign an order. At the slowest speed, 0° and 55° cannot register any collisions and are therefore the best there. This is no longer the case at the next two speeds, where 20°, 45° and 20°, 35°, respectively, perform best. At the highest speed tested, collisions are the lowest at 0° and 35°.

When looking at the calculation steps and path length, no extreme swings occur. The values increase with the increase of the angle but not significantly, cf. figure 4.7.

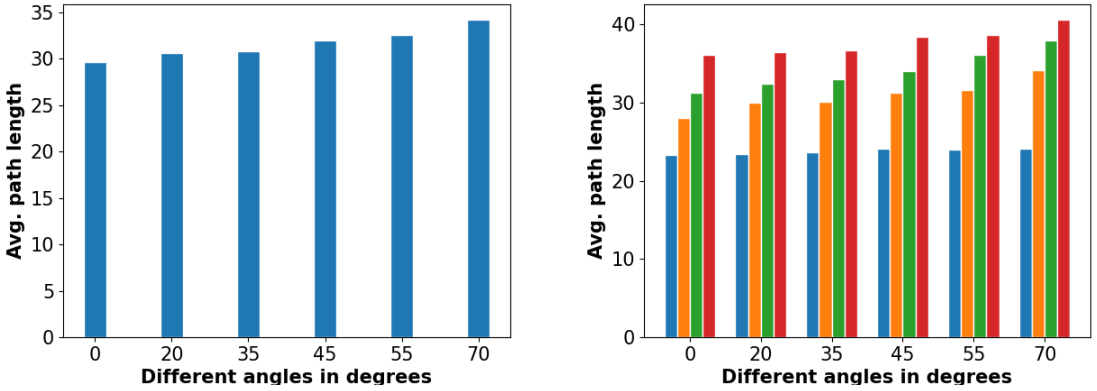
### 4.2.4 Improved APF

In the last step, the Improved APF extension is tested using the same procedures. However, the scaling term $k_v$ is now varied. The other parameters are $k_r = 5$ and $d_0 = 3$. The results can be seen in the graphs in figure 4.8.

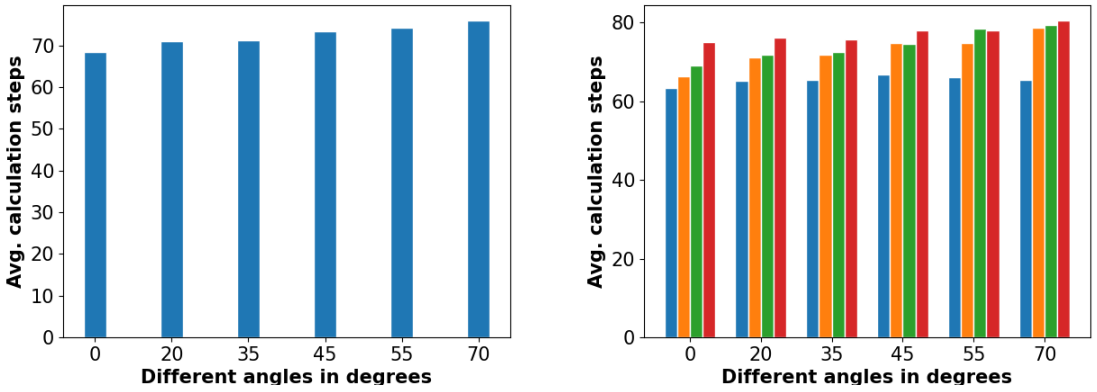
The number of collisions does not seem to improve significantly by changing the term compared to not using this extension, i.e. $k_v = 0$. The lowest average collision is at $k_v = 1$, but closely followed by $k_v = 0$. The other values have about the same number but are significantly higher than the first two. When dividing the velocities, one can see that although a clear order of the values with respect to the collisions was recognizable when looking at the whole average, this is not so easy. However, the tendency that $k_v = 1$ causes the least collisions remains, only at the lowest obstacle speed this value works the third best.




(a) Average collisions

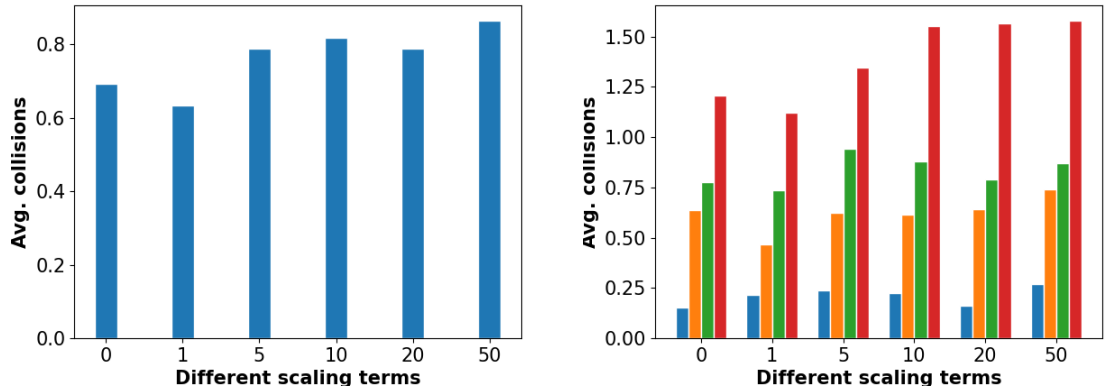



(b) Average path lengths

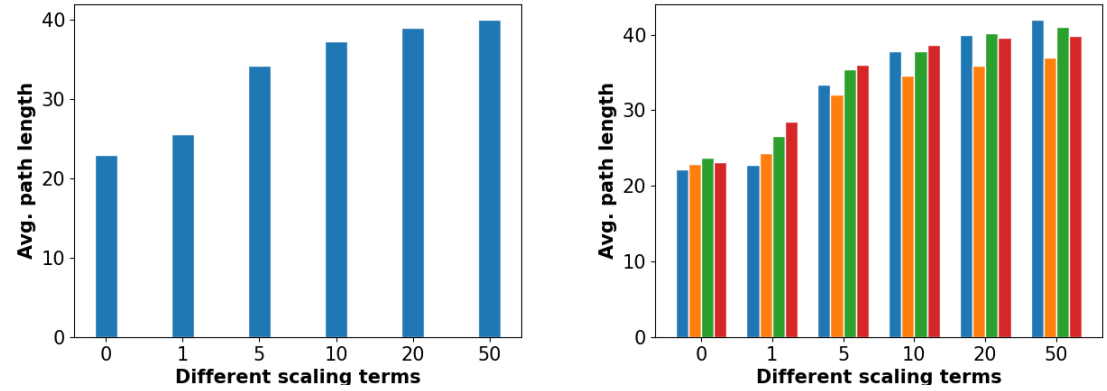



(c) Average calculation steps

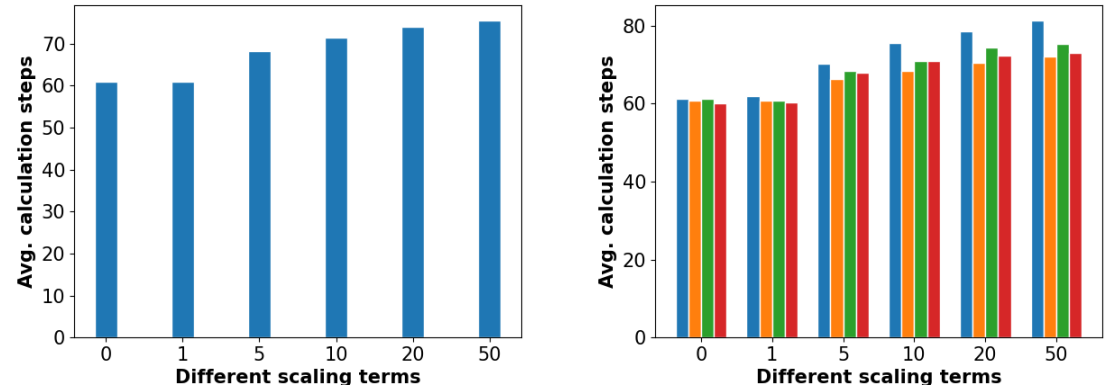
Figure 4.7: Results for the rotational APF when changing the rotation angle. Left: Average results for the entire test run for different rotation angles. Right: Average results for the entire test run for different rotation angles and separated in the different obstacle speeds. Different colors represent the different speeds (i.e. blue=0.5, orange=1, green=1.5, red=2).




(a) Average collisions




(b) Average path lengths




(c) Average calculation steps

Figure 4.8: Results for the improved APF when changing $k_v$. Left: Average results for the entire test run with different values for $k_v$. Right: Average results for the entire test run with different values for $k_v$ and separated in the different obstacle speeds. Different colors represent the different speeds (i.e. blue=0.5, orange=1, green=1.5, red=2).

When examining the calculation step and the path length, it is noticeable that when increasing the scaling term, these also increase constantly. When looking at the individual obstacle speeds, however, it is also observable that these values are close to one another at low scaling terms. At higher values, however, there are two striking deflections. With the path length the lowest value lies with the obstacle speed $v = 1$ and with the calculation steps the high values stand out with the lowest speeds.

## 4.3 Interim Evaluation

After the different models were first tested in the theoretical environment, a certain tendency of the different parameters already shows up. If the different models are now tested in a more detailed robot environment with an approximate equal environment size, these parameters can provide a good first insight.

Note that the rotation angle of 35° did not perform the best, but since the angle of 0° would match with the Forward Projection APF , it was decided to use this angle, since it best reflects the characteristics of its model.

Another point that has not been mentioned yet is the limitation of $d_0$ to 3 during testing and here, as well, although increasing this value resulted in fewer collisions. This is because in real robots, obstacle detection is achieved, for example, by a LIDAR sensor with a certain maximum range. Three meter is a good and justified assumption for such a sensor, because the built-in sensor of the Turtlebot3 has also this range.

| Model | $k_r$ | $d_0$ | future count | rotation angle | $k_v$ |
|---|---|---|---|---|---|
| classic apf | 5 | 3 | / | / | / |
| forward apf | 5 | 3 | 3 | / | / |
| rotational fapf | 5 | 3 | 3 | 35° | / |
| improved apf | 5 | 3 | / | / | 1 |

Table 4.1: Resulting parameters for the different models from the analyses of theoretical simulation.

# 5 Gazebo Simulation

Based on the results of the preceding theoretical tests, the different models will now be tested in the aforementioned Gazebo robot environment. It shall be tested in how far the models influence the collision avoidance as well as how applicable the parameters from section 4.3 are.

## 5.1 Settings

In contrast to the previous simulation, which assumed very strong simplifications of the environment and the robot, the simulation software Gazebo is now used. This is a powerful 3D simulation software for testing different types of robots. In it, different types of robots can be simulated realistically. Thus, the Turtlebot3 can be simulated and tested to see how well it would behave with the different models in the real world. To control the robot, the Robot Operation System (ROS) is used, just as with a physical robot. This provides a large software environment and serves as a kind of middleware to enable the exchange between different programs and the robot.

In this simulation, the Turtlebot3 with its more restricted freedom of movement is used. It starts at (-10,0) and the target is again at the coordinates (10,0). The obstacles are now real objects and can hit the robot and thus change its path. The objects are now moving in a 5x5 rectangle around the center at (0,0). The goal will be reached, when the robot enters an area three meters near (10,0).

As described in section 2.2, after the direction vectors have been calculated from the different models, they have to be converted into the control commands of the robot. This is done according to the equation presented. The direction vectors from the potential fields are again limited with a maximum length of 1. This results in a maximum linear velocity of the robot of 1 m/s according to the formula. Note that this robot speed is only possible in simulation. If working on a real robot, limiting the vectors even more in order to suit the maximal velocity constraints would be necessary.

It turned out that the robot with strongly changing direction vectors from the potential fields gets problems with moving. To make the robot movements more fluid, the result vector of the last calculation is added to each calculation step: $v_{used} = v_{calculated} \cdot 0.9 + v_{previous} \cdot 0.1$

Due to this maximum speed, two different obstacle speeds are observed in the following tests. First, the obstacles will also have a speed of 1 m/s. Thus, it would be possible for the robot to simply drive away from the obstacles if it detects them early enough. This is no longer possible with the second speed because the obstacles then move at twice the speed. Each model was tested 20 times at these speeds.

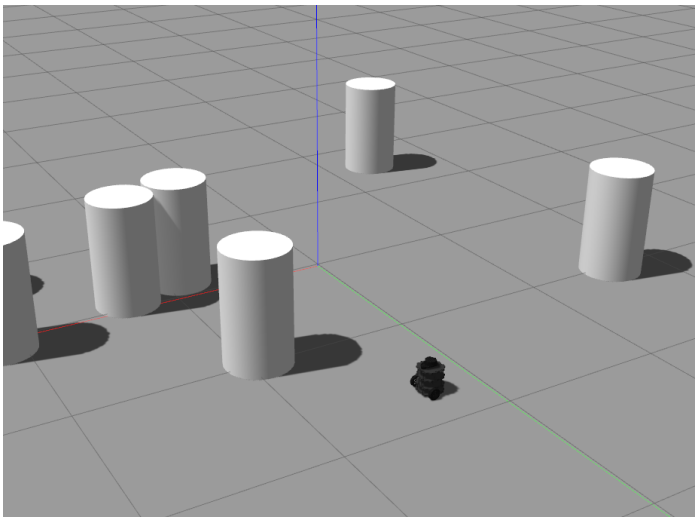


Figure 5.1: Section of the simulation environment with obstacles as cylinders and turtlebot3 on the right.

## 5.2 Results

First, the obstacle speed v=1 was tested with the different models (results shown in table 5.1). Here it was shown that the classic APF had the most collisions with an average of 0.9 collisions per run. Clearly the least collisions were achieved by the improved APF with an average of 0.3. In between are the values of the other two models, which are very close to each other with 0.55 and 0.6. However, the forward APF is ahead with fewer collisions.

| Model | avg. collision | avg. path length [m] | avg. time [s] |
|---|---|---|---|
| classic apf | 0.9 | 32.48 | 76.05 |
| forward apf | 0.55 | 25.75 | 59.71 |
| rotational fapf | 0.6 | 29.00 | 70.84 |
| improved apf | 0.3 | 25.16 | 55.70 |

Table 5.1: Gazebo simulation results for obstacle speed = 1

The average path length and the average time show a similar pattern. The improved APF not only travels the shortest routes (25.16 m) but also achieves them in the shortest time (55.70 s). The

forward algorithm drives only about 0.6 m longer routes but takes ≈ 4 seconds longer. The values of the rotational APF with an average length of 29 meters and a time of 70.84 seconds are significantly behind. Even more behind are the values of the classic model with another ≈ 3.5 meters plus and more than 5 seconds.

| Model | avg. collision | avg. path length [m] | avg. time [s] |
|---|---|---|---|
| classic apf | 3.1 | 30.09 | 69.93 |
| forward apf | 2.45 | 32.12 | 74.51 |
| rotational fapf | 3.1 | 31.40 | 74.57 |
| improved apf | 1.75 | 25.53 | 55.83 |

Table 5.2: Gazebo simulation results for obstacle speed = 2

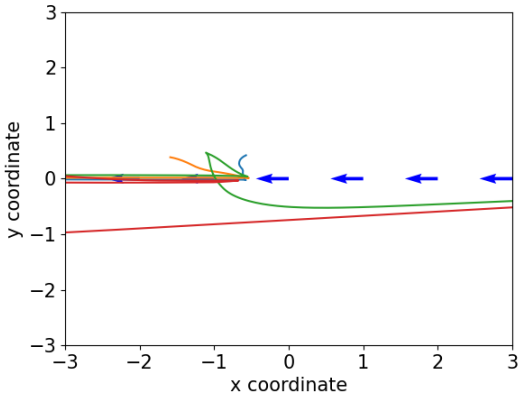
Even when increasing the speed of the obstacles to 2 m/s (see table 5.2), the improved APF remained clearly at the top with 1.75 average collisions compared to the other models. The number of collisions increased significantly compared to the slower speed, but the path length and time remained almost the same. For the other models, the number of collisions also increased remarkably. The values for the classic and the rotational APF are even the same this time with 3.1. The forward APF achieved a lower value with 2.45 average collisions, but this time it had to travel an average of 6.5 m longer routes and therefore took about 15 seconds longer. The values of the rotational APF also increased by only 2.5 m and 4 seconds compared to the slower speed. In contrast, the path length and time of the classic model decreased by 2.4 m and 6.1 s, respectively.

## 5.3 Additional Tests

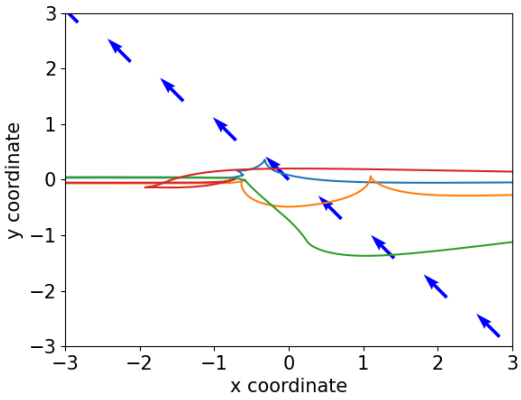
In addition to the tests above, three situations were considered individually for each model, in order to gain even more precise information about their different behaviors.

There is now only one obstacle on the map which moves from three different directions with 1 m/s to the center, starting from (0,-8), (5.66, -5.66) and (8,0). The robot starts for all models from (-5,0) and, since all models use the same attraction field, would collide with the obstacle exactly at the center point (0,0) without consideration of the obstacle. The routes traveled can be viewed in figure 5.2, and the calculated path lengths in table 5.3.

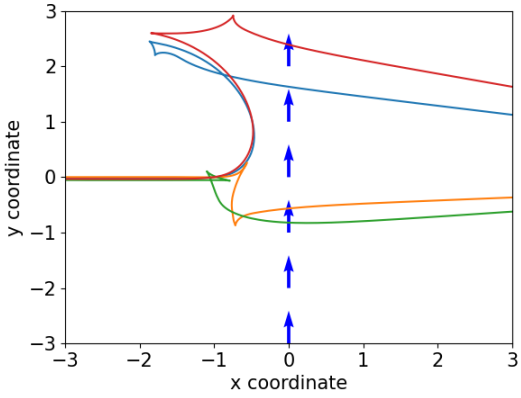
When the obstacle comes from the front (see 5.2a), both the classic and the forward APF collide. The other two succeed in driving collision-free, as both manage to drive out of the obstacle's path of movement. The improved APF, however, has to drive several meters in front of the obstacle and thus moves away from the target, making the path more than 10 meters longer.




(a) Paths when the obstacle starts from (8,0)




(b) Paths when the obstacle starts from (5.66,-5.66)




(c) Paths when the obstacle starts from (0,-8)

Figure 5.2: Section of the paths of the different models when an obstacle is on a collision course with the robot. The blue arrows reflect the movement of the obstacle. The different colors represent the different models (blue=classic APF, orange=forward APF, green=rotational forward APF, red=improved APF).

In the theoretical collision with a 45° angle, all models manage not to collide (cf. figure 5.2b). The classic and improved APF are both pushed backwards and only continue in a straight line when the obstacle is out of reach. The other two make an arc to the right, i.e., against the path direction of the obstacle. It should be noted that the rotational APF makes a larger arc, but after the obstacle has passed, it continues in a straight line towards the target. The forward APF does not succeed in this and must first realign itself, but since the driven arc is smaller they still have the same path length. The classic model has driven about the same distance, only the path of the improved APF is about 2 meters longer.

| Model | 180° | 45° | 90° |
|---|---|---|---|
| classic apf | / | 12.46 | 16.74 |
| forward apf | / | 12.63 | 13.56 |
| rotational fapf | 13.83 | 12.78 | 13.23 |
| improved apf | 24.40 | 14.50 | 16.93 |

Table 5.3: Average path lengths of the different models if an obstacle would be on collision course with the robot from different directions. Direction of movement of the obstacle characterized by its angle to the x-axis. No length is specified for a non-collision drive.

In the last of these tests shown in figure 5.2c, the obstacle came from the right and again all models succeeded without collision. Here, the behaviors of the classic, forward and improved APF are initially very similar. All three start with a left turn, but the forward model aborts this maneuver and drives the opposite direction and arrives as well as the rotational APF to the right of the theoretical collision point. The other two models finish their left turn after the obstacle is out of reach and the robot has completed a 180° turn. After that, they also drive directly towards the target. Since these two models were affected by the obstacle much longer, their path is also about 3.5 meters longer than the others.

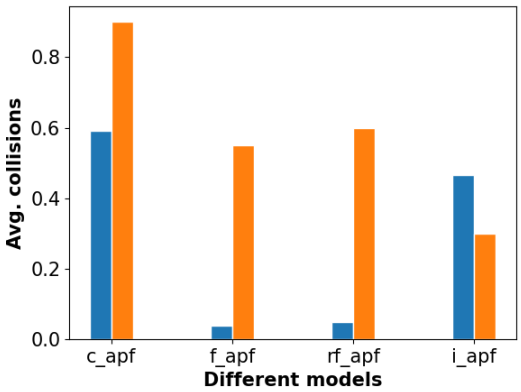
# 6 Evaluation

To finally evaluate the different models, the first step is to look at the average collisions of these models in the different simulations as shown in figure 6.1a and 6.1b. It quickly becomes apparent that in the two simulations used, different models have the fewest collisions. In the purely theoretical one, the forward APF and the rotational forward APF work best by far. In the more realistic gazebo simulation, however, the improved APF works best in terms of collisions. In both cases, however, the classic model clearly stands out as the worst, which is a very strong indication that this model is not suitable in a moving environment.

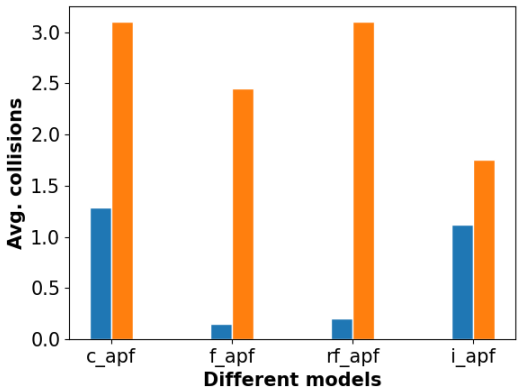
In general it is noticeable that the values in the two simulations differ greatly from one another. This is mainly due to the fact that in the theoretical model there were no movement restrictions on the robot, which meant that the robot was able to dodge faster and more effectively. The improved APF also performs particularly poorly in this one, since the calculation steps were not continuous enough and the robot therefore had too many erratic position changes, which resulted in incorrect relative velocities that did not correspond to the desired calculations. Therefore, this model is rather unsuitable in such a simplified simulation, although the results were nevertheless better than those of the classic model.

The biggest problem of the classic model is that it reacts too late to moving obstacles. Only when these are at a relatively short distance, an evasive maneuver is started. If the robot could move in any direction, it could still avoid many collisions. Regarding the Turtlebot, however, this is not really the case in real applications and a lot of time has to be spent braking and realigning. Simply increasing the effective field of the potential field would not solve this problem either, since no significant collision reduction could be observed (cf. figure 4.5a) and problems like local minima or the target not being reachable anymore became more and more present.

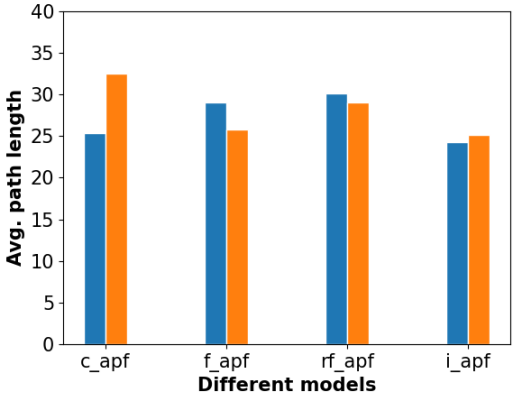
The forward APF now tries to solve these problems by extending the potential field only in the direction in which it is important. This results in a very strong collision reduction, which is especially noticeable in the theoretical simulation but also in Gazebo, see figure 6.1. Because of the larger repulsive fields of the obstacles a strongly swirled field is created, especially with multiple obstacles, which cannot be well reflected by the limited movements of the real robot. For this reason, the results in Gazebo are more similar to those of the classic model at higher speeds.



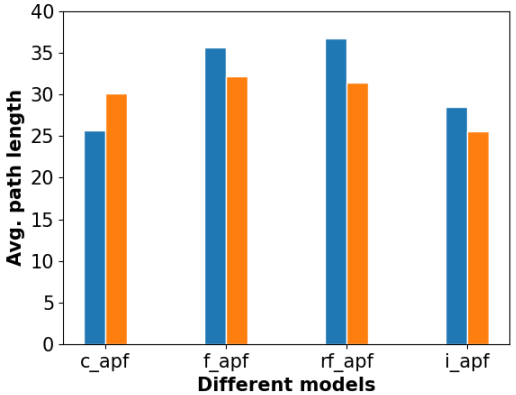

(a) Average collisions for the obstacle velocity = 1




(b) Average collisions for the obstacle velocity = 2




(c) Average path length for the obstacle velocity = 1




(d) Average path length for the obstacle velocity = 2

Figure 6.1: Average collisions and average path lengths for each model and different obstacle velocities in the different simulation tested. The results from the theoretical simulations are shown in blue and those from the gazebo simulation are in orange. Note the different scaling of the y-axis in the two graphs a) and b).

The same is true for the rotational APF which is the extension of the forward APF by rotation of the vectors. This change of rotation does not provide significant improvements. Rather, the results worsen in each category. This is due to two primary reasons, one is that the repulsion from the obstacle's line of motion is slightly weakened by the rotation, which causes the robot's maneuvers to become too slow in some places and a collision to occur. Secondly, the complete rotation of the field around an object also has the disadvantage that if an object is behind the robot and although no collision would take place the robot is now virtually pulled to the obstacle by the repelling field of the obstacle, i.e., away from the actual target.

However, the additional Gazebo tests in the section 5.3 show the potential of the rotational APF. Unlike the other models, this model specifies a direction in which the robot should steer. This

direction is specifically chosen so that the robot should steer behind the obstacle and is thus less influenced by it and the route becomes shorter. This could be seen when the obstacle came from the side of the robot. Also, when the obstacle came from the front, this was the best performing model because it came the fastest out of the path and the influence radius of the obstacle. The other models had a kind of moving local minima in this situation, since the obstacle was exactly on the axis of movement to the target and could only be solved if it was fast enough to drive away from the obstacle and there was an inaccuracy in the position of the robot and so could be slowly pushed out of the way.

As already mentioned, the used calculation method of the theoretical simulation does not fit optimally to the Improved APF. The results of the Gazebo simulation, which is clearly ahead of the other extensions, are all the more indicative. Not only could the number of collisions be reduced but also the path length is smaller. Unlike the other models, this model only really avoids obstacles that are on a collision course. Objects that move away from the robot, for example, are ignored, so that there is less interference with the potential field and the robot is less likely to deviate from the direct route that results only from the attraction field. In addition, an evasive maneuver can be executed more quickly and more effectively, since it is calculated more directly whether a collision is taking place and the robot does not first have to get into the artificial field of the pre-projections of the obstacle, which are significantly weaker further away from the obstacle than closer to it.

# 7 Conclusion and Outlook

In this work, the path planning concept of artificial potential field was investigated in a dynamic environment. It was suspected that this model is rather impractical in such an environment, so three extensions based on it were also investigated. A wide variety of investigations were carried out in a very simplified simulation and also in a very realistic Gazebo environment where the Turtlebot3 was used.

As supposed, the classic model, which gives good results in a static environment, turns out to be rather unsuitable. It lacks in particular the foresight of the moving obstacles, so that these are recognized too late and evasive maneuvers are usually no longer successful. All enhancements were able to reduce the number of collisions, but it was the improved APF that stood out the most. Not only did it excel with low collisions in the realistic Gazebo simulation, but it was also able to do so in much shorter routes. The forward APF and rotational APF could not achieve these low collision numbers, mainly due to the given limitations of the robot's movements. However, these two models could work much better with a faster and more agile robot. Another problem was that too many obstacles create a varying potential field, which could make it unusable in densely populated areas such as crowds or street traffic. However, it also turned out that with a few obstacles, the rotational APF could work better than the others because it can drive with more target orientation due to the rotation field of the obstacles.

At this point, this matter would have to be investigated further. The next step would be to see whether these results can also be achieved in the real world with the robot used. However, it should be noted that if the environment is chosen differently, the various parameters of the models would have to be adjusted. The maximum speed of the robot would also have to be adjusted, as this would be different. However, this could be achieved by scaling all vectors of the potential fields to the length of the maximum speed of the real robot.

When trying to further optimize the models, a possible approach would be to combine the different models presented. If the rotational APF had the property of the improved APF to focus only on a few obstacles that are on a collision course, this could result in an even safer model that not only delivers lower collision numbers, but also proactively drives past the obstacles to the target and is not only reactively influenced by them. Investigating the aforementioned issues will have a great

impact on how path planning can become more effective, not only in the artificial simulated, but also in the real world.

# List of Figures

# List of Tables

# Bibliography


[1] B. Brito, B. Floor, L. Ferranti, and J. Alonso-Mora, *Model predictive contouring control for collision avoidance in unstructured dynamic environments*, 2020. arXiv: 2010.10190 [cs.RO] (cit. on p. 2).

[2] F. Hegedüs, T. Bécsi, S. Aradi, and P. Gáspár, "Model based trajectory planning for highly automated road vehicles," *IFAC-PapersOnLine*, vol. 50, pp. 6958–6964, Jul. 2017. DOI: 10.1016/j.ifacol.2017.08.1336 (cit. on p. 2).

[3] M. S. Mohamad Hashim, T.-F. Lu, and H. Hassan Basri, "Dynamic obstacle avoidance approach for car-like robots in dynamic environments," Dec. 2012, pp. 130–135, ISBN: 978-1-4673-3032-9. DOI: 10.1109/ISCAIE.2012.6482083 (cit. on p. 2).

[4] A. Javaid, "Understanding dijkstra algorithm," *SSRN Electronic Journal*, Jan. 2013. DOI: 10.2139/ssrn.2340905 (cit. on p. 2).

[5] M. Psotka, F. Duchoň, M. Roman, T. Michal, and D. Michal, "Global path planning method based on a modification of the wavefront algorithm for ground mobile robots," *Robotics*, vol. 12, no. 1, 2023, ISSN: 2218-6581. DOI: 10.3390/robotics12010025. [Online]. Available: https://www.mdpi.com/2218-6581/12/1/25 (cit. on p. 2).

[6] Z. Xie and P. Dames, "Drl-vo: Learning to navigate through crowded dynamic scenes using velocity obstacles," *IEEE Transactions on Robotics*, vol. 39, no. 4, pp. 2700–2719, 2023. DOI: 10.1109/TRO.2023.3257549 (cit. on p. 2).

[7] O. Khatib, "Real-time obstacle avoidance for manipulators and mobile robots," *The International Journal of Robotics Research*, vol. 5, no. 1, pp. 90–98, 1986. DOI: 10.1177/027836498600500106. eprint: https://doi.org/10.1177/027836498600500106. [Online]. Available: https://doi.org/10.1177/027836498600500106 (cit. on p. 3).

[8] S. elia nadira, R. Omar, and C. K. N. Hailma, "Potential field methods and their inherent approaches for path planning," vol. 11, pp. 10 801–10 805, Jan. 2016 (cit. on p. 3).

[9] Y. Koren and J. Borenstein, "Potential field methods and their inherent limitations for mobile robot navigation," vol. 2, May 1991, 1398–1404 vol.2. DOI: 10.1109/ROBOT.1991.131810 (cit. on p. 6).

[10] Muhammad Sarim Mehdis, *Mapping & Path-Following For a Two-Wheeled Robot*, https://medium.com/@sarim.mehdi.550/mapping-path-following-for-a-two-wheeled-robot-b8bd55214405, Online; accessed 29 August 2023, 2020 (cit. on p. 6).

[11] J. GULDNER and V. I. UTKIN, "Tracking the gradient of artificial potential fields: Sliding mode control for mobile robots," *International Journal of Control*, vol. 63, no. 3, pp. 417–432, 1996. DOI: 10.1080/00207179608921850. eprint: https://doi.org/10.1080/00207179608921850. [Online]. Available: https://doi.org/10.1080/00207179608921850 (cit. on p. 6).

[12] A. Batinovic, J. Goricanec, L. Markovic, and S. Bogdan, "Path planning with potential field-based obstacle avoidance in a 3d environment by an unmanned aerial vehicle," in *2022 International Conference on Unmanned Aircraft Systems (ICUAS)*, 2022, pp. 394–401. DOI: 10.1109/ICUAS54217.2022.9836159 (cit. on p. 12).

[13] X. Fan, Y. Guo, H. Liu, B. Wei, and W. Lyu, "Improved artificial potential field method applied for auv path planning," *Mathematical Problems in Engineering*, vol. 2020, pp. 1–21, Apr. 2020. DOI: 10.1155/2020/6523158 (cit. on p. 14).

# Declaration of Academic Integrity

I hereby confirm that this thesis, entitled

*Evaluation and Enhancement of Artificial Potential Fields*
*for Path Planning in Dynamic Environments*

is solely my own work and that i have used no sources or aids other then the ones stated. All passages in my thesis for which other sources, including electronic media, have been used, be it direct quotes or content references, have been acknowledged as such and the sources cited. I am aware that plagiarism is considered an act of deception which can result in sanction in accordance with the examination regulations.

---

Niklas Hellmann, Münster, October 19, 2023

I consent to have my thesis cross-checked with other texts to identify possible similarities and to have it stored in a database for this purpose.

I confirm that I have not submitted the following thesis in part or whole as an examination paper before.

---

Niklas Hellmann, Münster, October 19, 2023