TUNING PERCEPTION AND MOTION PLANNING PARAMETERS FOR MOVEIT! FRAMEWORK

STEFAN GRUSHKO[1], ALES VYSOCKY[1], VYOMKESH KUMAR JHA[1], ROBERT PASTOR[1], ERIK PRADA[2], LUBICA MIKOVA[2], ZDENKO BOBOVSKY[1]

[1] VSB – Technical University Ostrava, Faculty of Mechanical Engineering, Department of Robotics, Ostrava, Czech Republic

[2] Technical University of Kosice, Faculty of Mechanical Engineering Department of Mechatronics,

Kosice, Slovak Republic

DOI : 10.17973/MMSJ.2020_11_2020064

stefan.grushko@vsb.cz

This paper performs a benchmark of main parameters of perception and planning available in MoveIt! motion planning framework in order to identify parameters the most affecting the overall performance of the system. The initial benchmark is performed on a virtual simulation of UR3 robot workspace with a single obstacle. The performance is measured by means of successful runs, path planning and execution durations. The results of the benchmark are processed and, based on the results, three parameters are chosen to be optimized using Particle Swarm Optimization. The optimization of the parameters is performed for the same motion planning problem as presented in the first benchmark. In order to test the performance of the system with optimized parameters, four more benchmarks are performed using the simulated and real robot workspace. The results of the benchmarks indicate improvements in most of the measured indicators.

KEYWORDS

Motion planning, perception, MoveIt, benchmark testing, manipulators, tuning, optimization

1 INTRODUCTION

Currently, many manufacturing processes have proven to be difficult to fully automate either because the task requires human-level perception and manipulation capabilities not yet achievable by robots or because they must be performed near human workers. To overcome this difficulty, a robot and human can collaborate in a shared workspace to perform manufacturing tasks [Vysocky 2016]. However, ensuring high efficiency and safety during this collaboration requires the robot to be able to avoid interference with the human and potential obstacles. Motion planning is, therefore, an indispensable skill for robots. Automatic replanning of robot trajectory is a complex task involving processing data from a sensor system usually consisting of multiple depth cameras, as well as utilising a fast solution for path search in the free space. Motion planning algorithms attempt to generate trajectories that are both feasible and optimal based on performance criteria that may vary depending on the task and environment.

High-level motion planning framework MoveIt! [Sucan 2013] is widely used for developing robotic applications requiring manipulation and automatic motion planning. MoveIt! allows to quickly and easily set up a perception pipeline for processing data obtained by depth cameras monitoring the workspace of the robot for further use during collision-free motion planning. The workspace environment represented by 3D point cloud data is continuously processed by MoveIt! creating a 3D occupancy map of the workspace – OctoMap [Hornung 2012] is generated from the filtered point cloud. Trajectory planning is done within this memory-efficient representation of the environment. During configuration of MoveIt! for a particular task, the user can set up and change multiple parameters for each of the motion planners as well as parameters of perception, such as the resolution of OctoMap. Many parameters can drastically change the performance of the system, and their actual optimal parameters highly depend on the environment, manipulation task and the robot itself. Since some of the parameters are interrelated, setting optimal values for the parameters requires complicated manual tuning.

The motivation for this paper is to conduct a basic benchmarking of main perception parameters and to evaluate an automatic optimisation of the parameters using an evolutionary optimisation method. The paper aims to provide insight into the influence of the main perception parameters available for configuration in MoveIt! on overall performance during motion planning and an approach to tuning some of these parameters. In addition to the testing of the perception parameters, for the defined motion planning problem, the best performing planner is chosen from 23 planning algorithms available in MoveIt!. The influence of each parameter is measured using a benchmark conducted on a virtual simulation of UR3 robot workspace. The configuration of the simulation model is set to closely match that of a real workspace that is used in further benchmarks of optimised parameters. Metrics for comparing the influences of the parameters on the overall system performance is chosen, and measured data are processed. Subsequently, based on processed data, the best performing planner is chosen and are selected three parameters for optimisation using an evolutionary optimisation method - Particle Swarm Optimization (PSO) [Kennedy 1995, Eberhart 2001]. After performing the optimisation of the parameters, the performance of optimised parameters is evaluated in four benchmarks which include two benchmarks conducted a real robot system. Lastly, the results and observations made during benchmarks are discussed, and a conclusion of the paper is provided.

2 BACKGROUND

Robot Operating System (ROS) is a robotics software platform [Quigley 2009]. It offers a distributed framework of processes that enables parts of the system to be individually designed and loosely coupled at runtime. ROS contains services, ready to use libraries and tools for the development of different types of robotic applications.

ROS is often used with high-level motion planning framework MoveIt! [Sucan 2013]. The framework is often used for a wide range of industrial and service robotic applications requiring automatic motion planning and is by default configured with Open Motion Planning Library (OMPL) [Sucan 2012]. This library consists of state-of-the-art sampling-based motion planners. MoveIt! uses OMPL planners to create a path to solve the defined motion planning problem.

Sampling-based motion planning algorithms avoid explicitly describing the entire configuration space of the robot - instead, a collision detection algorithm is used to probe the configuration space to determine whether a configuration state is in free space. The sampled configuration states are connected in order to find a feasible motion plan. This type of motion planners is widely used due to its success in finding feasible paths in high dimensional tasks. Typically, they have probabilistic completeness, which means that if a solution exists, it will be found (with an increasing number of iterations), but if not, the algorithm can run for an infinite time. Sample-based planners can be classified into two types: multi-query and single-query planners.

The approach of multi-query planners implies that at the pre-processing phase a roadmap representing connectivity of configuration space is generated (by iterative collection and testing non-collision states) and later multiple path search requests can be processed on its base by searching for a specific path by linking individual segments available in the roadmap. The pre-processing phase is required for multi-query methods, where it takes most of the computational time. Using the same roadmap again will decrease the computing time. However, the same map can only be used in a static environment.

Single-query planners do not create a roadmap representing the whole free space but generate a new tree-like graph each time they search for a solution. These tree graphs grow towards each other to link the initial and the goal configurations. Single-query methods are much faster, but the path found by the algorithm usually contains a lot of unnecessary curves and requires smoothing.

A brief overview of 23 OMPL motion planners available in the current version of MoveIt! (0.9.17) is provided in the works of Meijer *et al.* [Meijer 2017]. They investigated the performance of OMPL planners in a series of empirical studies, which have shown comparative effectiveness of planners used with multiple types of manipulators performing a set of grasping tasks. The benchmark data provided in their paper shows that the performance of the planners can slightly vary for different robots and task environments. The measurements were conducted in a simulated environment and did not utilise the perception pipeline of MoveIt!, hence no effect of perception parameters was tested.

A study performed by Burger *et al.* [Burger 2017] addressed the problem of automatic algorithm configuration for the case of motion planning algorithms available in the MoveIt! framework. Researchers proposed a tool based on Sequential Model-based Algorithm Configuration [Hutter 2011] to optimise the parameters of five motion planners on specific problems with the goal of improving the performance of the planners in terms of planning duration and solved runs. The performance of the planners was benchmarked in a series of simulated motion planning problems using two industrial manipulators; however, no cameras were used to monitor the surrounding of the robots during the tests, thus the effects of the perception settings were not investigated. Cano *et al.* [Cano 2016] in their work addresses the problem of automatic optimal configuration of distributed ROS system in terms of CPU utilisation and data flow from each of the optimised ROS nodes. The researchers first investigated the correlation between the setting of one critical parameter of each ROS node, its performance and CPU utilisation. Next, based on these relations, the optimal settings for each node of ROS graph was chosen. Research group of Cano *et al.* [Cano 2018] further extended the optimisation task to evaluate and compare the efficiency of four optimisation methods for tuning the parameters of two motion planning algorithms available from OMPL. The compared optimisation methods included: random sampling (as reference), AUC-Bandit approach [Alvaro 2010], Bayesian Optimisation [Shahriari 2016] and sequential optimisation based on Random Forest model [Hutter 2011]. Motion tasks were defined for a manipulator in the presence of obstacles in a simulated planning environment.

Nevertheless, neither of these studies included measuring the influence and optimising the parameters of perception of the surrounding, with respect to the overall performance of a system. An example of such parameter is the accuracy of the representation of perceived surrounding utilised for motion planning (parameter of OctoMap resolution in case of MoveIt!). Optimising these parameters requires setting up the acquisition of 3D data from a simulated environment and incorporating it into perception pipeline of the MoveIt! in order to bring the motion planning task closer to real conditions. Settings of the perception pipeline and planners expose a high number of possible parameter combinations which represents a large search space. Manually finding the best parameter configuration within this search space is tedious and time-consuming. Thus, an approach for an automatic search for optimal parameters for a particular planning task and environments is the main target of this work.

In this paper, we use both Gazebo [Koenig 2004] simulation and real UR3 robot workspace for performing the benchmarks. The workspace is monitored by a single depth camera RealSense D435. UR3 robot has 6 degrees of freedom, a working radius of 500 mm and a maximum payload of 3 kg. Stereo depth camera RealSense D435 has a depth image resolution of 1280x720 and can provide depth images at 90 frames per second with a maximum range of 10 m. There is no limit on the number of cameras observing the same object simultaneously since the cameras do not interfere with each other. The real and simulated workspaces of UR3 robot used in benchmarks are shown in Figure 2.

Particle Swarm Optimization [Kennedy 1995, Shi 1998, Eberhart 2001] is a population-based evolutionary optimisation method initially developed by Eberhart and Kennedy inspired by the real-world example of the behaviour of bird flocks. Swarm (flock) consists of particles (representing solutions for the defined problem) and each particle moves through the search space with a velocity, which is regularly updated by the particle's previous best performance and by the previous best performance of the swarm.

3 BENCHMARK OF PERCEPTION AND PLANNING PARAMETERS

As a benchmark environment (Benchmark 1) for the parameters, a Gazebo simulation of the robot workspace was used. The simulation model closely resembles the real robot workspace used in further benchmarks of the optimised parameter values. The UR3 robot is equipped with a gripper and placed on a table. The workspace is monitored by a single simulated RealSense D435 depth camera (see Figure 1), 3D point cloud data gathered from the simulation environment is published into ROS topic using Gazebo plugin.

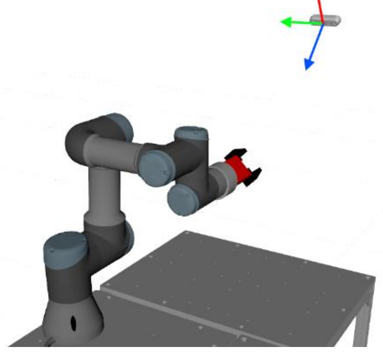

Figure 1. Location of the depth camera

The parameters of the virtual camera were set to match that of the real one. 3D point cloud data of the robot environment is continuously processed by MoveIt! so that the points that belong to the robot's body and its static periphery are removed from the cloud accordingly to the meshes defining their collision volumes. The initial end-effector (Home) position is located above the table in Home position of the robot. During the benchmark, the task of the robot was to consecutively move to goal positions A, B, back to A and return to Home position (see Figure 2). Goal positions are defined by joint angles. A simple cylinder placed on the table represented a movable obstacle on the path of the robot. The obstacle blocks the direct joint movement between the poses. Gazebo simulator is configured to use ODE physics engine.

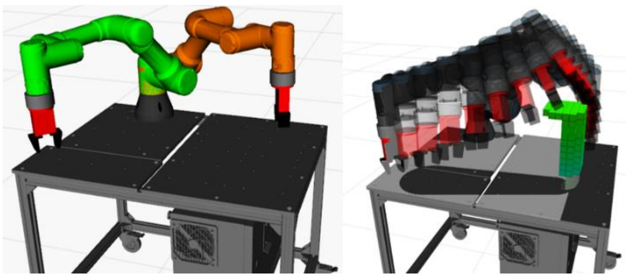

Figure 2. Robot movement during benchmarks: A and B goal positions (left); visualisation of movement around the obstacle (right)

3.1 Tested parameters

The performance influence of the following perception and planning parameters was tested during the benchmark:

- **Octomap resolution** – specifies the resolution at which the representation of the workspace is maintained in a 3D occupancy map (defined in meters). Changing this parameter can influence not only planning time but also the length of the planned movements.
- **Point subsample** – defines the degree of reduction of the density of the point cloud during the generation of 3D occupancy map. This parameter can significantly change the time required to update the map and manifests the most in a dynamic environment with moving obstacles. An excessive decrease in the value of this parameter can lead to a very sparse point cloud, which in turn will result in generating a sparse occupancy representation of obstacles. Sparse representation of obstacles in an occupancy map can cause the motion planning algorithm to find solutions in unnecessary proximity to these obstacles or even unfeasible path plans.
- **Workspace limitation** – when enabled, the available workspace for motion planning was constrained by the boundaries of the table on which the robot is located.
- **Collision mesh** – a model provided as collision volume of the robot. This parameter can drastically influence the performance because collision checking is performed multiple times during the path planning procedure. The simplified collision mesh reduces the number of polygons from 37686 to 2058 comparing to the default mesh of UR3 robot.
- **Goal joint tolerance** – defines the maximum permissible deviation of the joint position of the robot at which the pose found during motion planning is accepted as the goal pose.
- **Kinematics solver** – defines the kinematics solver to be used during motion planning. The default kinematics solver defined by MoveIt! is KDL Kinematics plugin suitable for general use with robots representing serial kinematic chains. Specific solvers, such as, UR3 Kinematics plugin, are explicitly compiled to be used with a specific robot.
- **Longest valid segment fraction** – defines the fraction of the robot's configuration space that, given the robot is not currently in a collision, it is assumed the robot can travel while remaining collision-free. If an edge between two nodes of a roadmap is less than this fraction of the configuration space, then no collision check will be performed along the edge between the nodes.
- **Max update rate** – defines the maximum update rate at which the OctoMap representation will be updated.
- **Path simplification** – enables simplification of the post-processing of the motion plan. It works by removing unwanted states and vertices from the obtained path.
- **Planner** – motion planning algorithm from the list of available planners in OMPL. This parameter was chosen for the testing in order to find the best performing parameter for the particular task environment used in the experiments. Meijer *et al.* **[Meijer 2017]** in their work conducted an extensive parameter selection for the available OMPL planners, so for use in benchmarks the planners' parameters described in their paper were used. BFMT and LBTRRT planners resulted in errors for the defined motion planning problem and will not be used during the benchmark.

For all parameters, the performance benchmarks were conducted and compared in two configurations: default and modified value. The default and modified values of the parameters for the benchmark are provided in Table 1.

Parameter	Initial value	Tested value(s)
Point subsample	1	150
Workspace limitation	Enabled	Disabled
Collision mesh	Default high poly model for UR3 37686 polygons	Simplified low poly model for UR3 2058 polygons
Goal joint tolerance (radians)	1e-05	0.01
Kinematics solver	KDLKinematics – generic kinematics solver	UR3Kinematics – UR3-specific kinematics solver
Longest Valid Segment Fraction	0.005	0.0005
Max update rate (Hz)	Unlimited	6
OctoMap resolution (meters)	0.02	0.005
Path Simplification	Enabled	Disabled
Planner	RRTConnect	SBL, EST, BiEST, ProjEST, KPIECE, BKPIECE, LBKPIECE, RRT, RRTConnect, PDST, STRIDE, PRM, LazyPRM, RRTstar, PRMstar, LazyPRMstar, FMT, TRRT, BiTRRT, SPARS, SPARStwo

Table 1. Tested parameters and their values

The test values were selected with regard to preliminary tests, which ensured that with all the specified parameter modifications (except for planners) the robot is able to perform at least one successful cycle.

3.2 Measured indicators

A total of *n=31* variants of the system settings (configurations *C = {c$_1$, …, c$_n$}*), including the initial, were compared. For each of the settings, 30 simulation cycles were performed: move from Home to A, from A to B, from B to A, from A to Home (this last motion is not measured). The first and the last movements were the least affected by the obstacle since its position did it restrain the robot's movements.

The goal of tuning the perception and planning parameters is to find such combinations of parameter values at which the time of planning and executing all movements will be minimised simultaneously with reaching the maximum ratio of successful attempts relative to the total number of attempts. Hence as metrics for the benchmark the following indicators of the performance were measured for the three first movements of the robot:

- Duration of motion planning – the time it takes for a planner to produce a plan. Low computing time is considered as high performance.
- Duration of execution of the planned movement – correlates with the length of the generated path. Low execution duration is considered as high performance.

Additionally, an indicator for cycle success ratio is measured - the success of the entire cycle expressed in terms of the percentage of total runs resulting in feasible paths without collision with the obstacle. The high success ratio is considered as high performance.

Total of *m=7* indicators *T = {t$_i$, …, t$_m$}* are measured for each run. To obtain reliable data, each setting was run 30 times for the given motion planning problem. The maximum motion planning time during all the benchmarks was limited to 5 seconds.

3.3 Results of benchmark

The benchmarking experiments are performed on a system with an Intel i7 2.80GHz processor, 16GB of RAM and Nvidia GeForce GTX 1070, Kinetic distribution of ROS running on Ubuntu 16.04, with Gazebo simulator 7.17 and MoveIt! 0.9.17.

Only successful cycles were used in the analysis: the planner should be able to find a solution for a defined problem, and the execution of the planned motion should be collision-free. The results of the benchmark are shown in Figure 4. The chart does not show values for planners which have not completed any successful cycle (either due to insufficient planning time or due to a collision of the robot with the obstacle). Only measurements shorter than 5 s are shown in the charts depicting the planning time.

Instead of analysing the metrics indicators individually, it was decided to bring all the data to a single scale and visualise in a single chart. Since measured duration and success ratio cannot be directly summed due to difference in the value ranges and opposite requirements, it was necessary to create a normalisation function *f($\bar{x}_{i,j}$, $\bar{x}_{i,j+1}$, … $\bar{x}_{i,m}$)* that would represent the overall success metric r_i of configuration c_i - total relative performance. The following transformation to relative performance (1) was used to normalise the measured planning and motion execution durations

$$r_{i,j} = \left(1 - \frac{\bar{x}_{i,j} - x_{j\,min}}{x_{j\,max} - x_{j\,min}}\right) * 100\,\%, \qquad j \in \{1, \ldots, 6\} \qquad (1)$$

where $\bar{x}_{i,j}$ - the average value of indicator t_j measured for configuration *c$_i$*,

$x_{j\,min}$ – the minimum value of t_j indicator measured for all configurations *C*,

$x_{j\,max}$ – the maximum value of t_j indicator measured for all configurations *C*.

The metric of the total relative performance r_i of c_i configuration is calculated as an average of the relative performances $r_{i,j}$ for all parameters (2)

$$r_i = \frac{1}{m}\sum_{j=1}^{m} r_{i,j} \qquad (2)$$

The normalisation function (1) was applied to all measured indicators for all configurations *C,* and the results are shown in Figure 3. The resulting graph shows the percentage performance of individual measured parameters for each measured variant c_i relative to other configurations (including the initial settings). Additionally, the chart shows the overall (total) performances of configurations calculated accordingly to equation (2).

Figure 5 illustrates the percent changes in overall system performance when setting modified values for each parameter compared to overall planning performance by default. The values were obtained by comparing the value of the total performance of each configuration with the performance of the initial configuration

$$d_i = r_i - r_1,\; i \in \{2..n\} \qquad (3)$$

where r_1 – total relative performance with the initial configuration.

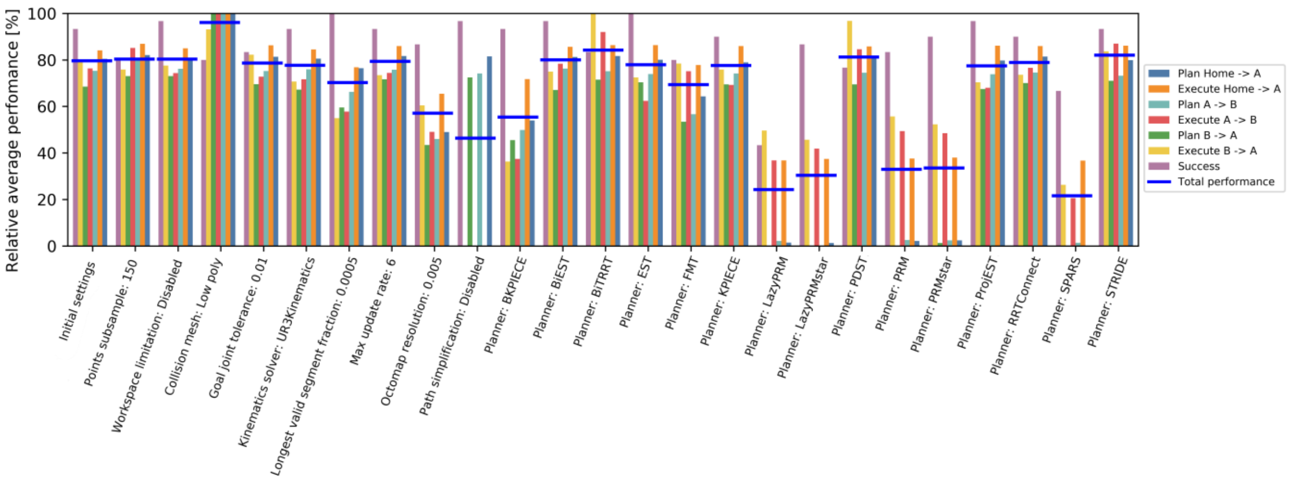

Figure 3. Relative performance of the measured configurations (overall performances are shown as blue marks), the higher is better

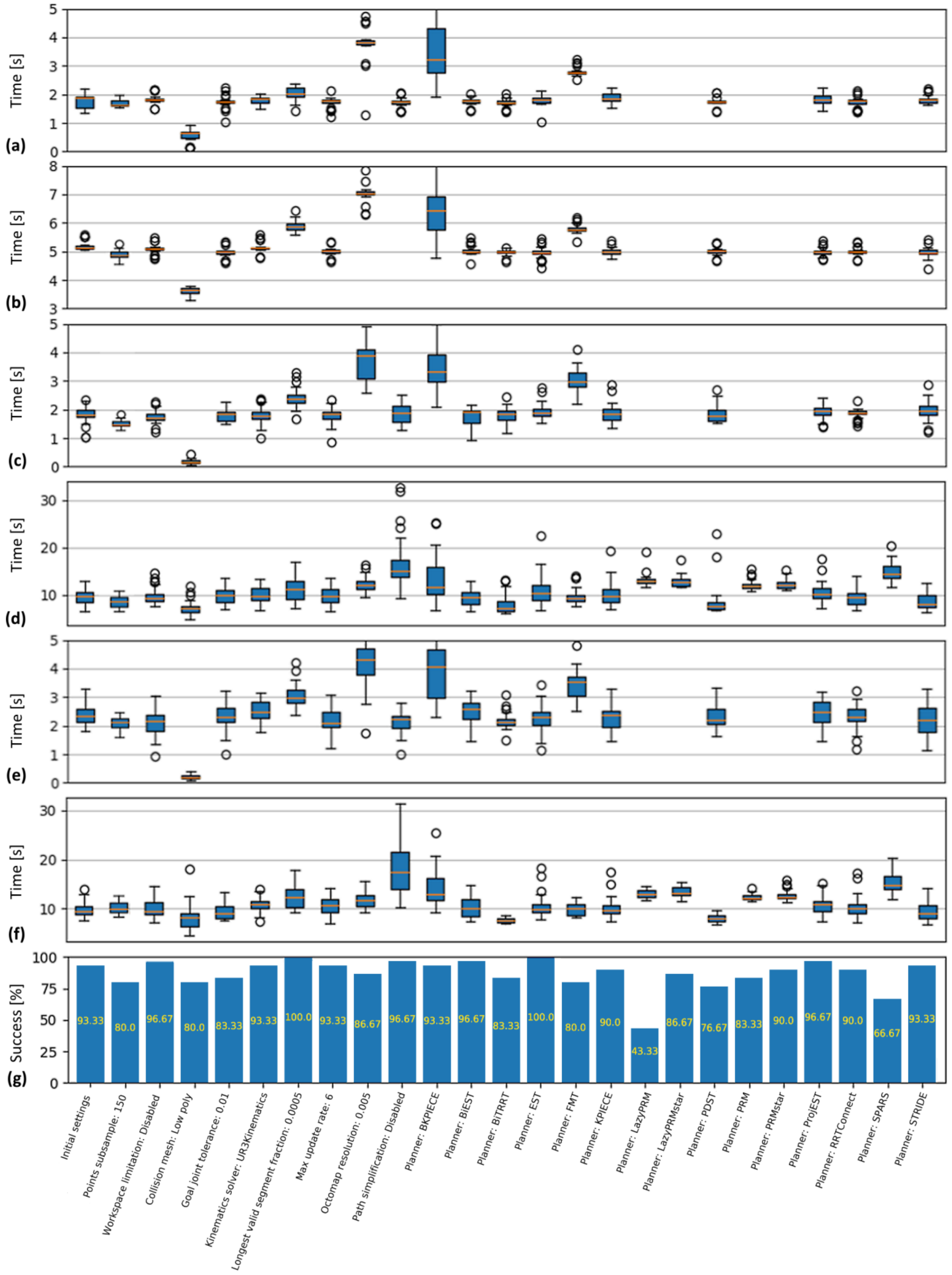

Figure 4. Parameters benchmark. (a) Planning time Home–A, lower is better; (b) Execution time Home–A, lower is better; (c) Planning time A–B, lower is better; (d) Execution time A–B, lower is better; (e) Planning time B–A, lower is better; (f) Execution time B–A, lower is better; (g) Percent of successful cycles, higher is better

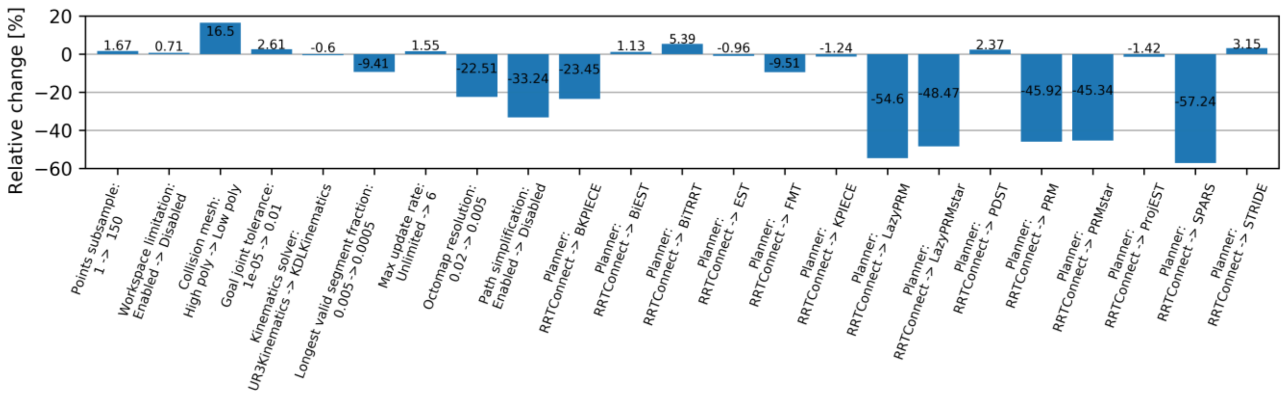

Figure 5. Change of the overall relative performance for each configuration. Positive values denote improvement; negative values denote deterioration

It can be seen from Figure 5 that certain parameters had a negative effect on overall system performance. For example, deactivated Path simplification resulted in a significant prolongation of the planned trajectories without much effecting the planning time. Disabled workspace limitation, as well as UR3-specific kinematics solver, do not show significant acceleration during planning. Drastically acceleration of the planning process can be observed with the simplified low poly collision mesh. Minor positive influences were registered for Points subsample, Goal joint tolerance, Max update rate and OctoMap resolution parameters. Modification of Longest valid segment fraction parameter caused a significant decrease in performance yet increasing the success ratio.

Considering the planners, success ratio over 90% was retrieved with EST, KPIECE, BKPIECE, PRMStar, ProjEST, RRTConnect, STRIDE. Nevertheless, considering the overall performance metric, the best performing planner for the defined motion planning problem was BiTRRT due to the lowest planning and movement durations, even though the success ratio of the planner was lower than 90%. While tracking the simulations, it was found that trajectories generated by BiTRRT are often much closer to the obstacle comparing to the trajectories of more reliable planners, leading to a higher chance of collision.

4 OPTIMISATION OF PARAMETER SETTINGS

This section describes an application of PSO optimisation algorithm to a task of finding an optimal combination of perception parameters leading to improvement in the overall performance of the system.

4.1 *Selecting parameters for optimisation*

Since the objective of optimising the combination of perception parameters that are interrelated is a task of optimisation of nonlinear function with unknown gradient, it was decided to use PSO algorithm. Based on the influence of individual parameters on the overall performance of the system analysed in the previous step, three parameters were chosen for optimisation:

- OctoMap resolution.
- Longest valid segment fraction.
- Points subsample.

These parameters are interrelated: for example, increasing the resolution of the OctoMap while reducing the density of the processed point cloud can result in an inaccurate definition of obstacle boundaries in the robot workspace. It was decided not to use the Goal joint tolerance parameter for optimisation because its value affects the accuracy of the resulting position sought by the motion planner. The optimisation was performed using a planner with the highest performance during the first benchmark - BiTRRT.

According to the previously chosen definition of the overall performance of the system, there are multiple objectives for the optimisation: objectives of minimising multiple duration parameters with different ranges and simultaneously maximising the success ratio. Instead of using the multi-objective optimisation [Gunantara 2018] approach required when multiple different functions must be optimised simultaneously, it was decided to use scalarising of the multi-objective optimisation problem to formulate a single-objective optimisation problem.

The goal of the optimisation algorithm is to find the optimal combination of perception parameters represented by the vector p* = [p1, p2, p3] ∈ S, which minimises the optimisation fitness function g(p) (4)

$$g(p^*) \leq g(p), \qquad \forall\, p \in S \tag{4}$$

where *S* is the search-space for *p* defined by the boundaries in Table 2 (lower and upper bounds).

As a scalarisation function, it was decided to use negated results of the normalisation function defined in equation (1) calculated using data measured during benchmark runs with the current particle. Additionally, unsuccessful attempts were penalised by multiplying the objective value of the fitness function outcome by a penalty factor b = 10. Without penalty factor, the algorithm tended to find only solutions with the shortest duration – parameters causing the robot to ram into the obstacle (leading to the shortest overall duration of the movements). As an environment for testing the same motion planning problem as during Benchmark 1 was used. Implementation of a modified PSO algorithm accordingly to [Shi 1998] was used. This implementation of the algorithm requires adjustment of three parameters: ω, c_1 and c_2. Inertia weight ω provides a balance between global and local explorations for each particle. The constants c_1 and c_2 are the scaling factors that pull each particle toward its personal best and global best positions, respectively. After a set of manual tests, the parameters set during the optimization as following: $\omega = 0.8$, $c_1 = 1.2$ and $c_2 = 1.4$. During optimisation, each solution (particle) was tested five times to average the results. Overall, two independent optimisations were made, each with 10 populations and 20 particles in each population.

Two candidate solutions (best results obtained by the particles during the runs of optimisation) are displayed in Table 2. It is very likely that the difference between obtained solutions is caused by randomisation that is in the principle of the sampling-based motion planner BiTRRT.

Parameter	Lower bound	Upper bound	Best candidate solution after the first run of optimisation	Best candidate solution after the second run of optimisation
OctoMap resolution - p_1	0.01	0.1	**0.0678**	0.0285
Points subsample - p_2	1	200	**125**	103
Longest valid segment fraction - p_3	0.001	0.01	**0.0056**	0.0071

Table 2. Results of optimisation

4.2 Benchmark of optimized parameters

In order to compare the performance of the system with the optimised parameters comparing with initial settings, four benchmarks were performed: 2 benchmarks in simulated environments (Figure 6) and 2 benchmarks with a real robot (Figure 7). The difference between the benchmarks is in different positions and shapes of the obstacles presented in the workspace of the robot. For the benchmarks the first candidate solution (see Table 2) was used p* = [0.0678, 125, 0.0059].

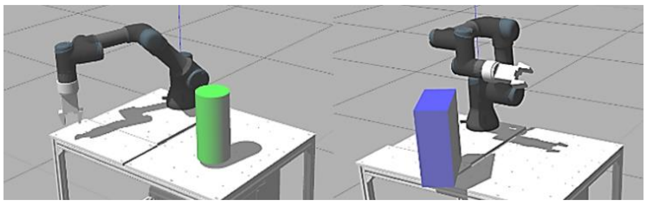
Figure 6. Benchmarks 1 (left) and 2 (right) in a simulated environment

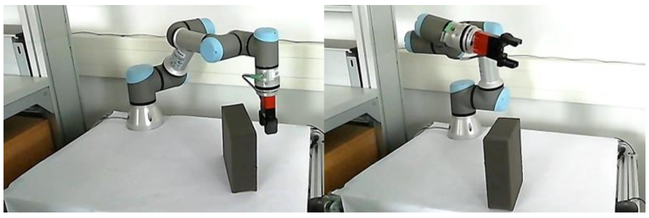
Figure 7. Benchmarks 3 (left) and 4 (right) on real robot

During each benchmark, system performance was compared for four configurations:

1) Initial configuration with BiTRRT planner, default high poly collision mesh. Parameters OctoMap resolution, Points subsample, Longest valid segment fraction are set to default values.
2) Configuration with optimised parameters (OctoMap resolution, Points subsample, Longest valid segment fraction) and BiTRRT planner.
3) Initial configuration with RRTConnect planner and default high poly collision mesh. Parameters OctoMap resolution, Points subsample, Longest valid segment fraction are set to default values.
4) Configuration with optimised parameters (OctoMap resolution, Points subsample, Longest valid segment fraction), BiTRRT planner and low poly collision mesh.

Configurations were compared in the following accordance:

- Configuration 1 (Initial, BiTRRT, High poly) with Configuration 2 (Optimised, BiTRRT, High poly) to assess only the influence of the optimised parameters.
- Configuration 3 (Initial, RRTConnect, High poly) with Configuration 4 (Optimised, BiTRRT, Low poly) to assess the total change of the indicators after applying all the adjusted parameters (including the low poly collision mesh).

To give reliable data on the performance of these configurations, each was measured 30 times for the given motion planning problem. The results of the benchmarks are shown in Table 3 and Table 4. Tables represent mean durations of planning and execution for each movement as well as the overall success ratio of the configurations.

Both benchmarks performed in the simulation using the optimised configurations show improvements in most of the measured indicators (durations of motion planning and execution). However, benchmark 2 (Table 3) shows a significant decrease in success ratio using optimised parameters with BiTRRT planner. After a more detailed analysis, it was found that in this particular case the setting of the OctoMap resolution and points subsample parameters led to an inaccurate mapping of the obstacle boundaries, which in turn led to a generation of trajectories in excessive proximity to the obstacle. It can thus be assumed that the reason is conducting of the optimisation process in the environment of the Benchmark 1, which shows the advantage of the optimised configuration comparing over the initial one. The parameters were optimised for a specific motion problem - the particular position of the obstacle.

	Real robot - Benchmark 1				Real robot - Benchmark 2			
	Initial BiTRRT High poly	Optimised BiTRRT High poly	Initial RRTConnect High poly	Optimised BiTRRT Low poly	Initial BiTRRT High poly	Optimised BiTRRT High poly	Initial RRTConnect High poly	Optimised BiTRRT Low poly
Plan Home-A [s]	1.780	1.533	1.599	0.373	1.804	1.592	1.865	0.194
Rel. change [%]	13.882		76.665		11.774		89.616	
Execution Home-A [s]	5.172	4.836	4.825	3.470	6.720	5.103	9.819	5.150
Rel. change [%]	6.499		28.089		24.069		47.553	
Plan A-B [s]	1.802	1.660	1.563	0.266	1.717	1.456	1.889	0.122
Rel. change [%]	7.877		83.005		15.232		93.558	
Execution A-B [s]	8.133	8.504	9.723	9.481	6.765	8.162	9.598	6.159
Rel. change [%]	-4.563		2.481		-20.656		35.828	
Plan B-A [s]	2.209	2.001	2.396	0.277	2.081	1.838	2.144	0.126
Rel. change [%]	9.414		88.440		11.680		94.119	
Execution B-A [s]	7.825	7.980	12.882	8.913	7.876	8.179	9.676	5.154
Rel. change [%]	-1.980		30.812		-3.849			46.734
Success ratio [%]	83.333	93.333	93.333	96.666	93.333	30.000	100.000	23.333
Rel. change [%]	10.000		3.333		-63.333		-76.667	

Table 3. Results of optimisation in benchmarks with simulation. Positive values denote improvement; negative values denote deterioration

	Real robot - Benchmark 3				Real robot - Benchmark 4			
	Initial BiTRRT High poly	Optimised BiTRRT High poly	Initial RRTConnect High poly	Optimised BiTRRT Low poly	Initial BiTRRT High poly	Optimised BiTRRT High poly	Initial RRTConnect High poly	Optimised BiTRRT Low poly
Plan Home-A [s]	1.880	1.877	2.245	0.336	1.848	1.834	1.964	0.282
Rel. change [%]	0.130		85.024		0.767		85.629	
Execution Home-A [s]	5.368	5.302	5.742	3.712	7.111	6.451	9.153	4.985
Rel. change [%]	1.242		35.354		9.289		45.542	
Plan A-B [s]	1.902	1.655	2.170	0.325	1.821	1.712	2.025	0.276
Rel. change [%]	12.958		85.043		5.978		86.385	
Execution A-B [s]	8.743	8.451	11.229	6.370	8.063	6.925	10.345	4.934
Rel. change [%]	3.341		43.268		14.108		52.304	
Plan B-A [s]	2.193	2.081	2.672	0.317	2.103	1.690	2.357	0.296
Rel. change [%]	5.119		88.126		19.647		87.443	
Execution B-A [s]	5.514	8.547	10.548	6.358	7.892	7.664	10.628	4.445
Rel. change [%]	-55.016		39.725		2.893		58.177	
Success ratio [%]	83.333	90.000	100.000	86.667	83.333	63.333	100.000	53.333
Rel. change [%]	6.667		-13.333		-20.000		-46.667	

Table 4. Results of optimisation in benchmarks with the real robot. Positive values denote improvement; negative values denote deterioration

Benchmark results with a real workplace (Table 4) are similar to the results of benchmarks conducted in simulated Gazebo environment. Almost all measured cycle parameters show improvement, although the success ratio indicator during Benchmark 4 showed a decrease due to more frequent robot collisions with an obstacle. Similar to Benchmark 2 (Table 3) it was found that the reason was an inaccurate mapping of the obstacle boundaries due to the settings of OctoMap resolution and points subsample parameters – the edge of OctoMap nodes did not cover the obstacle completely. It is necessary to be taken into account that the density and accuracy of the point cloud obtained by a stereo depth camera decreases with the increasing distance of the monitored object **[Sung 2019, Chuang-Yuan 2019]**. This corresponds with the fact that both benchmarks 2 and 4 used more distant obstacles than ones used during benchmarks 1 and 3. The observed decrease of success ratio is also likely to be caused by the use of BiTRRT planner, which already showed a reduction of value of this indicator during the first stage of the benchmarking (see Figure 3, Figure 4).

The best improvements of performance indicators along with the highest influence on the parameters were achieved for all benchmarks for Configuration 4 (with simplified low poly collision mesh). The use of optimised parameters without simplified collision mesh on average for all benchmarks led to a decrease in planning duration by 9.5% although the movement duration increased on average by 2.1%. In case of using optimised parameters along with simplified collision mesh, the planning duration was on average decreased by 86.9%, and the movement duration decreased on average by 38.8% (see Table 3, Table 4).

5 DISCUSSION

In this section, from the results of the benchmarks observations are made and discussed. The influence of the perception and planning parameters on the performance was studied by means of success ratio, motion planning and execution duration. The chosen total performance metric for comparing the configurations equally opt for all the indicators measured. Should be taken into consideration that this metric can be less suitable for applications requiring higher success ratio – for example collaborative workspace where safety is the most important factor, the metric can be changed in order to opt more for configurations with higher success ratio meaning a lower chance of collision of the robot with a human operator.

Considering the comparison of overall performance for available planners (Figure 3) the best performing planner for the defined motion planning problem was BiTRRT. Nevertheless, the success ratio for this planner was lower than 90%. A possible solution for increasing the success ratio is to increase obstacle padding - the required minimum distance to the obstacle during the planning of a trajectory. Currently available version of MoveIt! (release 0.9.17) does not allow configuring this parameter (or more precisely – the parameter does not affect the planning), but the same effect in a real robot could be achieved using an oversized collision model (padded mesh) of the robot. Despite the beneficial effect of the optimised perception parameters on the performance, the positive effect of the simplified collision mesh for the robot is significantly higher.

More investigation into parameters of particle swarm optimisation is needed in order to achieve a more reliable and faster optimisation process. The optimisation process can be carried out with more populations in order to achieve global best results for the given motion planning problem. Given lower success ratios in Benchmarks 2, 4 (Table 3, Table 4) it might be considered that the optimised parameters are not fully transferable to other motion planning tasks once the optimisation is performed for a specific task environment (in this case Benchmark 1). The optimisation should be carried out for multiple variations of obstacle positions and multiple modifications of the start and goal positions of the manipulator, so the optimised configuration will achieve better results for a wider range of tasks. A monitoring system consisting of multiple depth cameras can be used, in order to overcome the problem with inaccurate mapping of the obstacles in the workspace.

For this research, the conducted benchmarks did not consider dynamic obstacles. The result of the optimisation in a dynamic environment is likely to be different from the results acquired with the static ones, some of the perception parameters may show more influence on the overall performance of the system.

6 CONCLUSIONS

The paper presented benchmark data for main perception and planning parameters available for configuration in the motion planning framework MoveIt!. The influence of each parameter was measured during a benchmark conducted on a virtual simulation of UR3 robot workspace. Parameters' influences of the performance were studied by means of success ratio, motion planning and execution duration. Metrics for comparing the impact of the parameters on the overall system performance were chosen, and the measured data were processed. Based on processed data, BiTRRT planner was identified as the best performing planner for the defined motion planning problem. Additionally, considering the influence of individual parameters were selected three parameters for performing optimisation of their values using an evolutionary optimisation method - Particle Swarm Optimization. After performing the optimisation of the parameters, the performance of the optimised parameters was evaluated in 4 benchmarks, 2 of which were conducted a real robot system. As exhibited in Table 3, Table 4 the benchmarks shown comparative improvements in most of the measured indicators; however, the use of BiTRRT planner with optimised parameters has decreased the success ratio of the performed attempts.

The proposed method of optimisation of perception and planning parameters along with the evaluation of their influence on the overall system performance is useful for easing the hard task of manual tuning of these parameters in order to get better performance, thus minimising the amount of background knowledge required to use planning algorithms. The evaluation and optimisation are performed using MoveIt! framework considering it is ubiquitous use.

In our future research, we would like to investigate the option to implement a faster and more robust method for optimising the perception parameters given a wide variety of motion planning tasks, including environments with dynamic obstacles. Considering that potential users of the optimisation may be interested in finding optimal parameters in the shortest possible time, more intensive selection of PSO parameters [Rezaee 2013] must be performed since these parameters drastically change the time needed for the swarm to explore the parameter search space. Alternatively, other optimisation approaches as Bayesian Optimization [Shahriari 2016] or Sequential Model-Based Optimisation [Hutter 2011] can be considered regarding their efficiency in similar applications and potentially may use less time to find better configurations. However, results of benchmarks conducted for motion tasks different from the one where the optimisation was performed indicate that to obtain stable improvements for a wide range of applications, optimisation across multiple diverse tasks must be performed. Thus, a set of common motion tasks must be created, which should include typical industrial environments, simulating, for example, dynamic environment during human-robot collaboration, pick and place tasks. The environments should include multiple monitoring depth cameras to overcome the problem of the inaccurate mapping of the obstacles in the working environment. Given the environments and the typical motion planning tasks, it is also of use to perform optimisation of the visual coverage of the workspace by the cameras by adjusting their positions.

ACKNOWLEDGMENTS

This work was supported by the Research Platform focused on Industry 4.0 and Robotics in Ostrava Agglomeration project, project number CZ.02.1.01/0.0/0.0/17_049/0008425 within the Operational Program Research, Development and Education. This article has been also supported by specific research project SP2020/141 and financed by the state budget of the Czech Republic and VEGA 1/0389/18 Research on kinematically redundant mechanisms.

REFERENCES

[Alvaro 2010] Álvaro, F., Marc, S. & Michèle, S. Fitness-AUC Bandit Adaptive Strategy Selection vs. the Probability Matching one within Differential Evolution: An Empirical Comparison on the BBOB-2010 Noiseless Testbed. Proceedings of the 12th annual conference companion on Genetic and evolutionary computation. Portland, Oregon, USA, 2010, pp. 1535–1542, ISBN 9781450300735.

[Burger 2017] Burger, R., Bharatheesha, M., Eert, M. v. & Babuška, R. Automated tuning and configuration of path planning algorithms. IEEE International Conference on Robotics and Automation (ICRA). Singapore, 2017, pp. 4371-4376, ISBN 978-1-5090-4633-1.

[Cano 2016] Cano, J. et al. Automatic configuration of ROS applications for near-optimal performance. IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). Daejeon, South Korea, 2016, pp. 2217-2223, ISBN 978-1-5090-3762-9.

[Cano 2018] Cano, J. et al. Automatic Parameter Tuning of Motion Planning Algorithms. IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). Madrid, Spain, 2018, pp. 8103-8109, ISBN 978-1-5386-8094-0.

[Eberhart 2001] Eberhart, R. & Shi, Y. Particle swarm optimization: developments, applications and resources. Proceedings of the 2001 Congress on Evolutionary Computation. Seoul, South Korea, South Korea, 2001, pp. 81-86, ISBN 0-7803-6657-3.

[Gunantara 2018] Gunantara, N., Qingsong, A. A review of multi-objective optimization: Methods and its applications. Cogent Engineering, 2018, 5(1), ISSN 2331-1916.

[Hornung 2012] Hornung, A. et al. OctoMap: an efficient probabilistic. Autonomous Robots. 2012, 34(3), pp. 189–206, ISSN 0929-5593.

[Hutter 2011] Hutter, F., Hoos, H. & Brown, K. L. Sequential Model-Based Optimization for General Algorithm Configuration. Proceedings of the 5th international conference on Learning and Intelligent Optimization. Berlin, Germany, 2011, pp.507-523, ISBN 978-3-642-25565-6.

[Chuang-Yuan 2019] Chuang-Yuan, C. et al., Comparison of depth cameras for three-dimensional reconstruction in medicine. Proceedings of the Institution of Mechanical Engineers Part H Journal of Engineering in Medicine. Proceedings of the Institution of Mechanical Engineers, Part H: Journal of Engineering in Medicine, 2019, pp. 938-947, ISSN 0954-4119.

[Kennedy 1995] Kennedy, J. & Eberhart, R. Particle swarm optimization. Proceedings of ICNN'95 - International Conference on Neural Networks. Perth, WA, Australia, Australia, 1995, pp. 1942-1948, ISBN 0-7803-2768-3.

[Koenig 2004] Koenig, N. & Howard, A. Design and use paradigms for Gazebo, an open-source multi-robot simulator. IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). Sendai, Japan, 2004, pp. 2149-2154, ISBN 0-7803-8463-6.

[Meijer 2017] Meijer, J., Lei, Q. & Wisse, M. An empirical study of single-query motion planning for grasp execution. IEEE International Conference on Advanced Intelligent Mechatronics (AIM), Munich, Germany, 2017, pp. 1234-1241, ISBN 978-1-5090-5998-0.

[Meijer 2017] Meijer, J., Lei, Q. & Wisse, M. Performance study of single-query motion planning for grasp execution using various manipulators. 18th International Conference on Advanced Robotics (ICAR). Hong Kong, China, 2017, pp. 450-457, ISBN 978-1-5386-3157-7.

[Quigley 2009] Quigley, M. et al. ROS: an open-source Robot Operating System. ICRA Workshop on Open Source Software. Kobe, Japan, 2009.

[Rezaee 2013] Rezaee, J. A. & Jasronita, J. Parameter selection in particle swarm optimisation: A survey. Journal of Experimental & Theoretical Artificial Intelligence, 2013, 25(4), ISSN 0952-813X.

[Shahriari 2016] Shahriari, B. et al. Taking the Human Out of the Loop: A Review of Bayesian Optimization. Proceedings of the IEEE, 2006, 104(1), pp. 148-175, ISSN 0018-9219.

[Shi 1998] Shi, Y. & Eberhart, R. A Modified Particle Swarm Optimizer. IEEE International Conference on Evolutionary Computation Proceedings. IEEE World Congress on Computational Intelligence. Anchorage, AK, USA, USA, 1998, pp. 69-73 ISBN 0-7803-4869-9.

[Sucan 2013] Sucan, I. A. & Chitta, S., 2013. MoveIt!. [Online] Available at: http://moveit.ros.org

[Sucan 2012] Sucan, I. A., Moll, M. & Kavraki, L. The Open Motion Planning Library. IEEE Robotics & Automation Magazine, 2012, 19(4), pp. 72–82, ISSN 1070-9932.

[Sung 2019] Sung, A. M. et al. Analysis and Noise Modeling of the Intel RealSense D435 for Mobile Robots. 16th International Conference on Ubiquitous Robots (UR). Jeju, South Korea, 2019, pp. 707-711, ISBN 978-1-7281-3232-7.

[Vysocky 2016] Vysocky, A. & Novak, P. Human - Robot collaboration in industry. MM Science Journal, 2016, 2(6), pp. 903-906, ISSN 18031269.

CONTACTS:

Ing. Stefan Grushko
VŠB – Technical University of Ostrava
Faculty of Mechanical Engineering, Department of Robotics (354),
17. Listopadu 2172/15, Ostrava, 708 33, Czech Republic
+420 597 329 443, stefan.grushko@vsb.cz, robot.vsb.cz