applied sciences

MDPI

Article

Method for Robot Manipulator Joint Wear Reduction by Finding the Optimal Robot Placement in a Robotic Cell

Tomáš Kot [1,*], Zdenko Bobovský [1], Aleš Vysocký [1], Václav Krys [1], Jakub Šafařík [2] and Roman Ružarovský [3]

[1] Department of Robotics, Faculty of Mechanical Engineering, VSB-Technical University of Ostrava, 17. listopadu 2172/15, 708 00 Ostrava-Poruba, Czech Republic; zdenko.bobovsky@vsb.cz (Z.B.); ales.vysocky@vsb.cz (A.V.); vaclav.krys@vsb.cz (V.K.)
[2] Laboratory of Big Data Analysis, IT4Innovations, VSB-Technical University of Ostrava, 17. listopadu 2172/15, 708 00 Ostrava-Poruba, Czech Republic; jakub.safarik@vsb.cz
[3] Department of Production Devices and Systems, Slovak University of Technology in Bratislava, Ulica Jána Bottu č. 2781/25, 917 24 Trnava, Slovakia; roman.ruzarovsky@stuba.sk
* Correspondence: tomas.kot@vsb.cz

Abstract: We describe a method for robotic cell optimization by changing the placement of the robot manipulator within the cell in applications with a fixed end-point trajectory. The goal is to reduce the overall robot joint wear and to prevent uneven joint wear when one or several joints are stressed more than the other joints. Joint wear is approximated by calculating the integral of the mechanical work of each joint during the whole trajectory, which depends on the joint angular velocity and torque. The method relies on using a dynamic simulation for the evaluation of the torques and velocities in robot joints for individual robot positions. Verification of the method was performed using CoppeliaSim and a laboratory robotic cell with the collaborative robot UR3. The results confirmed that, with proper robot base placement, the overall wear of the joints of a robotic arm could be reduced from 22% to 53% depending on the trajectory.

Keywords: robot; manipulator; robotized workplace; robotic cell; optimization; wear

Citation: Kot, T.; Bobovský, Z.; Vysocký, A.; Krys, V.; Šafařík, J.; Ružarovský, R. Method for Robot Manipulator Joint Wear Reduction by Finding the Optimal Robot Placement in a Robotic Cell. *Appl. Sci.* **2021**, *11*, 5398. https://doi.org/10.3390/app11125398

Academic Editor: António Paulo Moreira

Received: 17 May 2021
Accepted: 7 June 2021
Published: 10 June 2021

1. Introduction

The design of robotized work cells or lines is a complex multidisciplinary task that is influenced by many external factors and conditions. During the designing process, it is necessary to make many decisions that greatly affect the resulting performance of the workplace and its properties, including the cycle time, velocity of individual parts, dynamic effects, vibrations, lifetime, ground area, and energy consumption.

Crucial for the design of a robotic cell is the selection of an appropriate industrial or collaborative robot and its placement in relation to other subsystems. The length of the designing stage of robotic cells is still shortening, which leads to the copying of existing layouts of workplaces with similar parameters and their adaptation to the actual requirements.

Any type of advanced optimization is almost impossible in this approach. However, this is the phase where it is possible to achieve some interesting savings of energy consumption, ground area, or the lifetime of some systems. The recent progress in complex simulation tools, such as Tecnomatix, Matlab—Robotics System Toolbox, CoppeliaSim (V-Rep), Gazebo, and Webots; methods for the creation of digital twins of the designed robotized workplace; and whole manufacturing lines [1] open up new room for the realization of complex multicriteria optimizations in the early stages of design.

Optimizations of robotized workplaces to achieve the reduction of running costs are currently mostly realized in systems that are already in operation. These typically include modifications of the end-point trajectory according to a chosen criterion, such as the manipulation time [2,3] or the overall productivity and running costs of the robot [4].

The optimization of the kinematic properties of the manipulator movement can also lead to the reduction of torques in individual joints and, thus, also the reduction of the overall energy consumption. For example, the authors in [5] used interval analysis to find the global minimum jerk trajectory of a robot manipulator in a joint space using cubic splines, resulting in a smoother and vibration-free robot movement.

A similar approach was proposed in [6], where the objective function included not only the integral of the squared jerk values but also the total execution time of the trajectory. The authors in [7] used the 12-phase sine profile for trajectories of a fast-moving robot, which leads to a more stable and accurate movement without sudden changes of torques, albeit at the cost of a slightly higher overall energy consumption. Trajectory optimization with the single goal of cycle time reduction using the chicken swarm optimization (CSO) method was described in [8].

Another optimization goal is the reduction of the workplace layout size [9] because the usable area of the factory building is extremely valuable. There are also some methods for the multicriteria optimization of robotic work cells or lines for energy consumption—even for lines with up to 12 robots, based on the Gurobi simplex method [10]. Another method of multi-robot cell optimization for time and energy reduction using a custom mechatronic model of the robots in Modelica/Dymola was described in [11]. The authors in [12] proposed a methodology for assembly line energy consumption optimization based on the implementation of energy-optimal trajectories, and, in [13], this method was improved by modification of the actuator brake release time for additional energy consumption reduction.

A systematic methodology for the on-site identification and energy-optimal path planning of an industrial robot is presented in [14] with a focus on a specific type of ABB industrial robot. Improved robot programming reduced the energy consumption compared to the built-in controller routines by up to 4%.

Further improvements and torque reduction for a robotized work cell can also be achieved by modification of the position and orientation of the robot base in relation to the optimized trajectory [15]. This can be interesting, especially for applications where it is not possible to modify the end-point trajectory and velocity for technological reasons (the application of adhesives, edging, welding, etc.). In these cases, it is necessary to create a complex simulation model—a digital twin of the workplace [16,17] and perform a multicriteria optimization.

The Concept of Robot Wear

Gearboxes are often mentioned as the component that is frequently responsible for a failure of rotary machines [18], including industrial robots [19]. The most critical components of a gearbox are the gears and bearings. The deterioration of a gear appears at the teeth [20,21], harmonic drive gears are damaged by fatigue fractures [22], and bearings are typically subjected to failures at the rolling elements or the inner or outer race [21]. The damage accumulates over time and is caused mainly by load (forces and velocity) [23,24], high temperature [25], bad lubrication [26], or manufacturing defects. The performance, accuracy, and lifetime of a robot relies on the good condition of these critical components [27].

The authors in [28] proposed a method for estimating the wear of a robot by monitoring the temperatures in the joints, while [29] suggested a more complex solution, where other parameters are also monitored in order to predict the lifetime of a multi-component system. Other common properties monitored to predict the lifetime of a machine include vibrations and noise [21]. A generic framework for predictive maintenance based on simulation models with degradation curves discovered from real data was proposed in [27]. Our approach, on the contrary, attempts to minimize wear by the use of a quite simple simulation in the design stage of a robotized workplace instead of monitoring an already existing one.

As mentioned above, the gearbox in the robot manipulator joint can be considered the most important source of the wear of the joint. The electric motor in the joint is subject to deterioration as well [30]. As far as the above-mentioned sources of wear are concerned, the most important is the load combined with movement velocity, because a higher velocity creates higher temperatures. This applies to both the gearbox and the motor. The influence of bad lubrication or manufacturing defects are almost impossible to anticipate or even simulate and, thus, are not considered in this work.

A typical robot joint contains structural bearings that provide the mutual rotational movement of the joints. However, obtaining the values of the reacting forces required to calculate the wear of these bearings is not an easy task, even when using a dynamic simulation. This requires an exact simulation model of the robot with a detailed representation of the inner parts and mechanisms, accurate values of the mass properties, acceptably realistic values of the friction coefficients, and a good dynamic simulation engine. Although it is usually possible to obtain 3D models of commercially available industrial and collaborative robots, and sometimes even simulation models prepared for common simulation systems, these models typically are simplified and do not contain the inner mechanisms of the arms.

On the other hand, wear of the gearbox and motor can be expressed as torque that the motor must generate (and that the gearbox must transfer to the joint) over some trajectory of motion. These values depend on physical quantities that are much easier to obtain from a simulation—the overall path of motion (kinematics) and the required driving torque required to achieve the given acceleration (dynamics).

The hypothesis of our research is that the lifetime of a robot in a robotized workplace can be improved in the design stage by designing the workplace (namely the location of the robot) in such a way to ensure lower wear of the robot joints.

2. Materials and Methods

To determine the optimal placement of a robot manipulator within a robotic cell with the goal of reducing and balancing joint wear, it is necessary to propose a suitable optimization criterion, the whole optimization process, and an experiment to verify the results.

2.1. Optimization Criterion

In this work, we consider only the wear of the robot drive chain (see the previous section). We will also restrict the following notation to rotational joints (which are much more common than translational joints in robotics) and to six degrees of freedom (the most common number in robotics); however, the principles can be applied in general.

An industrial or collaborative robot in a robotized workplace usually performs a limited set of movements. Typically, there is a given trajectory that the robot end-point should follow during the work cycle, and the robot joints are controlled using inverse kinematics to adhere to this trajectory. The idea of wear being caused by a torque acting over a trajectory corresponds with the concept of mechanical work, which can be expressed as

$$W = \int_{\phi_1}^{\phi_2} \tau d\phi, \tag{1}$$

where τ is the magnitude of the torque vector, ϕ is the angle of rotation about the vector representing the joint axis, and ϕ_1 and ϕ_2 are the starting and ending angles of rotation, respectively.

The integral (1) is path-dependent and the mechanical work is defined as the change of energy. Thus, if we assume $\phi_1 = \phi_2$, which is true for a closed-loop trajectory of the robot end-point (all individual joints have to start and end in the same angle of rotation), the resulting value of W would always be equal to zero. It is, thus, more convenient to express mechanical work as the integral of mechanical power over time

$$W = \int_{\phi_1}^{\phi_2} \tau d\phi = \int_{t_1}^{t_2} \tau \omega dt, \tag{2}$$

where ω is the angular velocity of motion and t_1 and t_2 are the starting and ending time, respectively. However, this integral (2) is still path-dependent, and the calculation would result in $W = 0$. Thus, it is necessary to introduce the absolute value of both the torque and the angular velocity

$$W = \int_{t_1}^{t_2} |\tau \omega| dt, \tag{3}$$

which allows us to calculate the work over the whole trajectory, while in fact, considering the trajectory divided into segments separated by the change of direction of movement or the change of the sign of τ. This modification moves away from the concept of the conservation of energy toward the concept of wear caused by mechanical work. The idea is that the drive chain of a robot joint is worn down even when the torque is negative (the motor is actively braking) or when the angle of rotation is decreasing or moving back.

The simulation is numerical with a definitive value of Δt (simulation step size) instead of dt. Therefore, the integral (3) is replaced by a sum

$$W = \sum_{i=0}^{n} |\tau_i \omega_i| \Delta t, \tag{4}$$

where $i = 0, 1, \ldots, n$ is the simulation step, τ_i is the instant torque, and ω_i is the instant angular velocity in the i-th simulation step. The mathematical meaning of the value W is shown in an example in Figure 1.

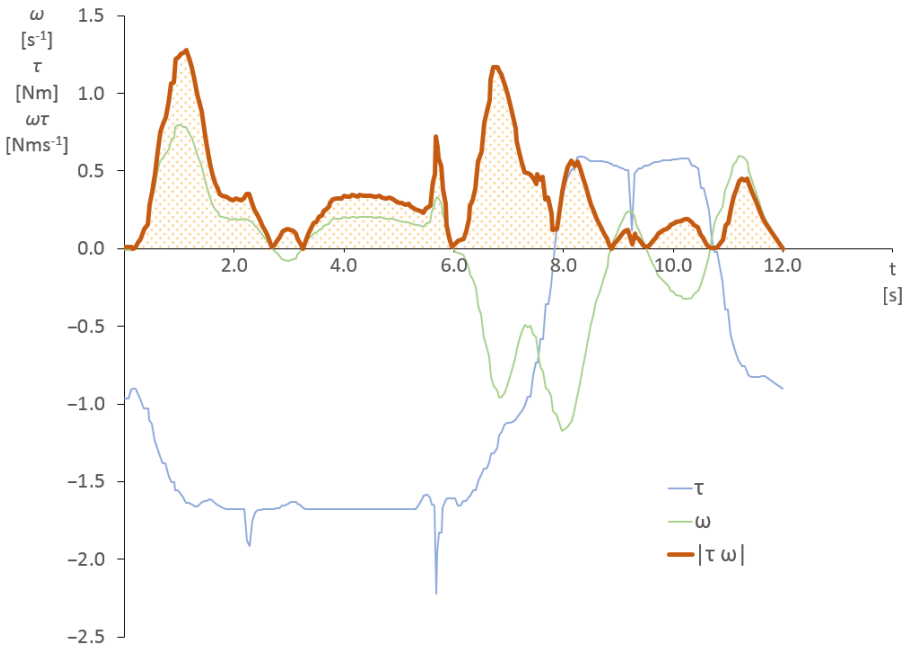


Figure 1. Example of the time progression of joint torque τ and angular velocity ω; the dotted area represents the meaning of the value W calculated as the integral of absolute value $|\tau\omega|$.

The value of W was calculated individually for each joint over the whole robot trajectory, yielding $W_1, W_2, \ldots, W_6$ for the most common number of 6 degrees of freedom.

We attempted to optimize two factors:

- to minimize the overall wear of all joints, and
- to balance the wear of all joints.

Therefore, it is necessary to consider the values W_j together. However, the joints of a robot manipulator are typically not equal from the mechanical point of view, and

their capabilities are different—the lower joints are larger and stronger, and the joints near the end-effector are lighter. The same magnitude of W could, in reality, mean much higher stress and wear for a smaller joint than for a larger one. We, therefore, introduce a variable called the relative wear factor w_j, which is calculated by normalizing the individual W_j values

$$w_j = \frac{W_j}{\tau_{m_j}\omega_{m_j}}, \quad w_j \geq 0, \tag{5}$$

where $j = 1, 2, \ldots, 6$ is the joint number, τ_{m_j} is the maximal permissible torque, and ω_{m_j} is the maximal angular velocity of the j-th joint (both values are specified by the manufacturer of the robot).

To minimize the overall wear of the robot, we chose to use the arithmetic mean of the relative wear factors of all joints and to find the minimal value of this mean,

$$A = \frac{1}{6}\sum_{j=1}^{6} w_j, \tag{6}$$

and balance is achieved by finding the minimal value of the standard deviation

$$\sigma = \sqrt{\frac{1}{6}\sum_{j=1}^{6}(w_j - A)^2}. \tag{7}$$

The chosen fitness function (optimization criterion) f_f places the same weight on both these factors; therefore,

$$f_f = \frac{A}{2} + \frac{\sigma}{2}. \tag{8}$$

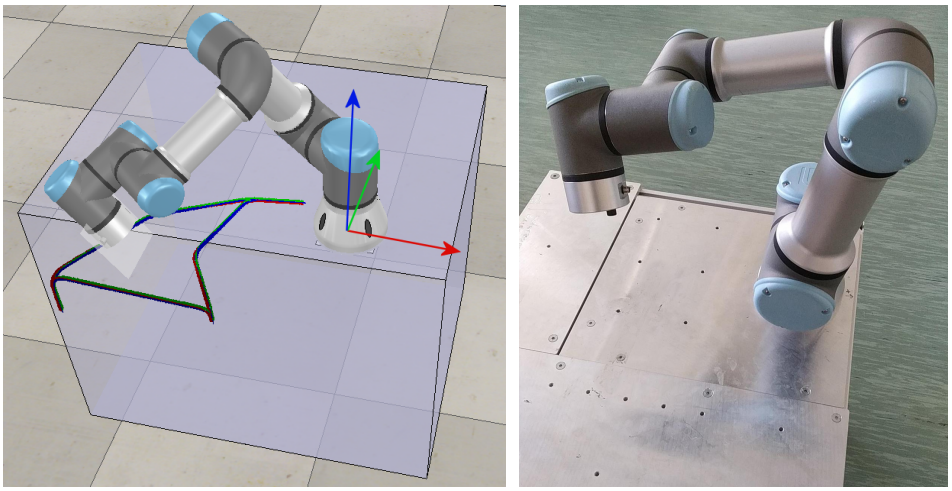
2.2. Optimization Process

The proposed method requires a simulation model capable of evaluating the movement of a robot manipulator through a given end-point trajectory while computing the values of joint angles and torques in individual time steps. This is possible, for example, in the popular robotic simulation system CoppeliaSim (formerly known as V-Rep), which was also chosen for our work.

The scripting capability of CoppeliaSim allows programming a robot's end-point movement through a given trajectory and time, and the built-in physics engine (i.e., Bullet 2.78) is able to check for collisions and evaluate joint torque values. The inverse kinematics (IK) are calculated using the integrated pseudoinverse IK solver. The CoppeliaSim API (Application Programming Interface) framework (*RemoteAPI*) can be used to remotely configure the simulation, which, in our case, includes particularly the process of changing the robot placement relative to the trajectory, as we are trying to determine the optimal placement of the robot. A custom application was written in Visual C++ for this purpose.

Due to the long simulation times in CoppeliaSim, especially in the cases when the IK calculation fails (the robot cannot reach some parts of the given trajectory), all valid positions of the robot relative to the given trajectory were pre-calculated in the C++ application and CoppeliaSim was used only to calculate the fitness function value in those positions. The valid locations were found using a simple kinematic simulation inside the C++ application, which can quickly verify the ability of the robot to fulfill a given task from the specific location, including collision checking.

The search space is represented as a discrete 3-dimensional grid with the spacing in all three dimensions equal to s = 0.03 m. The system returns the valid robot positions as a list of points, where each point represents the location of the center of the robot base (see the coordinate system in Figure 2a) in the workspace. Verification of the whole robot trajectory in this simulation system took approximately 1800-times less time than the same task in CoppeliaSim, and invalid robot locations were discarded even faster.



(**a**) (**b**)

Figure 2. The UR3 robot. (**a**) The simulation model in the CoppeliaSim environment with a path representing the end-point trajectory and a coordinate system in the center of the robot base. (**b**) The real robot used in the experiments.

For the reproducibility of the research, this supplementary custom simulation system is not necessary, as its main purpose is simply to shorten the overall simulation time. However, it is necessary to have an external application or a script in CoppeliaSim that successively places the robot in the locations from the above-mentioned search space (grid), executes the CoppeliaSim simulation, and calculates and stores the results.

The reason for choosing a 3-dimensional grid for the potential locations of the robot base instead of a 2-dimensional plane (representing the factory floor) is that the proposed method is intended to be as general as possible, and limiting the search to a 2-D grid could likely miss some interesting solutions. In reality, robots are commonly mounted on a stand, table or a console; therefore, the height of the robot can be chosen. However, the results can be easily limited to, for example, a 2-dimensional plane, if the actual application requires such a limit.

2.3. Experiment Setup

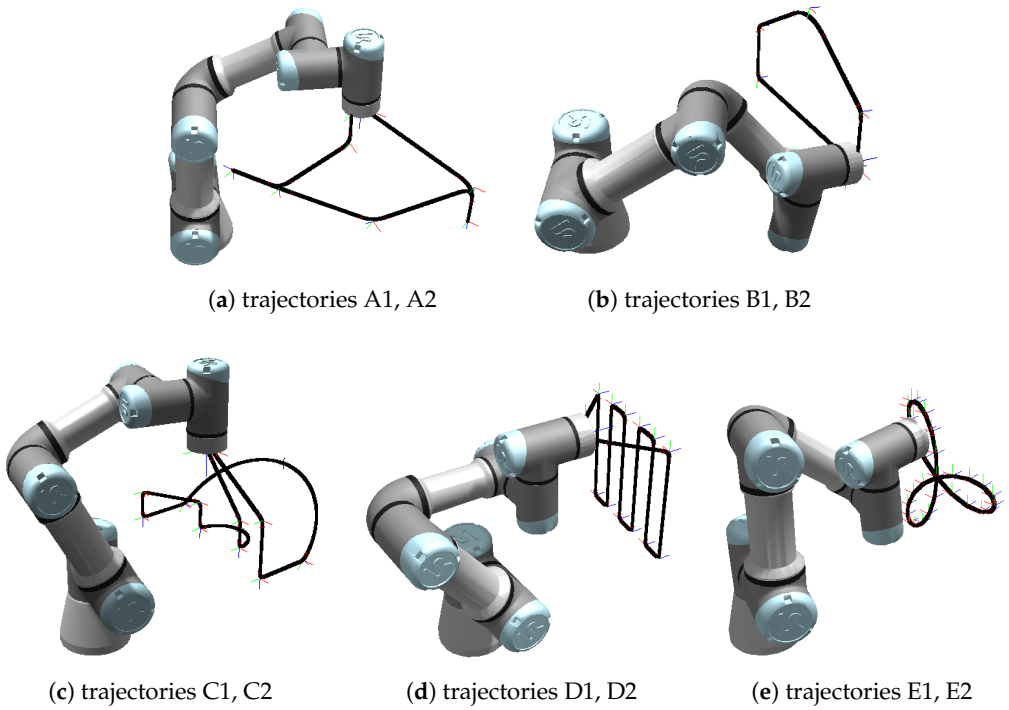
The proposed method was demonstrated and verified on a Universal Robots UR3 collaborative robot with 6 degrees of freedom (see Figure 2), first in a CoppeliaSim simulation configured according to the previous chapter, and finally also on a real physical robot. The values of the maximal allowed joint torque τ_{m_j} and joint angular velocity ω_{m_j} (5) for the UR3 robot are listed in Table 1.

Five testing movement paths of the robot end-point were demonstrated, each in two variants with different velocities of the end-point (0.1 and 0.2 m/s), giving a total number of ten trajectories labeled A1, A2, B1, B2, . . ., E2; where A–E is the path type, 1 stands for 0.1 m/s, and 2 stands for 0.2 m/s.

Table 1. The joint parameters for the UR3 robot; τ_{m_j} is the maximal permissible torque, and ω_{m_j} is the maximal permissible angular velocity of the j-th joint.

j	τ_{m_j} [Nm]	ω_{m_j} [s^{-1}]
1	56	π
2	56	π
3	28	π
4	12	2π
5	12	2π
6	12	2π

The trajectories are a combination of real use-cases and artificially created simple trajectories that contain various elements (linear segments of different lengths, circle arcs, sections with frequent speed, direction changes, etc.), see Figure 3. Various velocities of the end-point cause a difference in the solution of possible robot base locations, because every robot has some velocity limits for individual joints (see Table 1), and they also represent different cases regarding the proposed wear factor calculation.



(**a**) trajectories A1, A2 (**b**) trajectories B1, B2

(**c**) trajectories C1, C2 (**d**) trajectories D1, D2 (**e**) trajectories E1, E2

Figure 3. Visualization of the five testing robot end-point paths; the UR3 robot is shown as a scale reference.

The selected trajectories were of a smaller size, and there were no obstacles in the environment (except for the table that the robot was mounted onto), which leads to a larger number of different valid robot locations relative to the trajectory and, thus, also a larger data set of results.

The chosen robot UR3 is a small robot with a reach of only 500 mm and although using larger trajectories combined with several obstacles in the environment would be possible, the valid locations of the robot would be very limited and it would be difficult to make a statistical evaluation. In general, the proposed optimization method does not depend on the size of the trajectory. A longer trajectory, requiring a longer time to traverse, would produce larger values of the discrete integral (4); however, this is true for all robot locations relative to the particular trajectory, and thus the optimization remains valid.

In a real case, the simulation model would have to contain also all collision objects in the workplace, which could limit the valid robot locations considerably—the principle of the method is, however, still applicable. The concept of choosing the robot location is demonstrated on Figure 4 for the trajectory A1. The figure shows the grid of valid robot locations—each blue point represents a possible position of the robot base; the robot is able to reach the whole trajectory from each of these valid locations. For better clarity, the robot is shown in several selected locations.

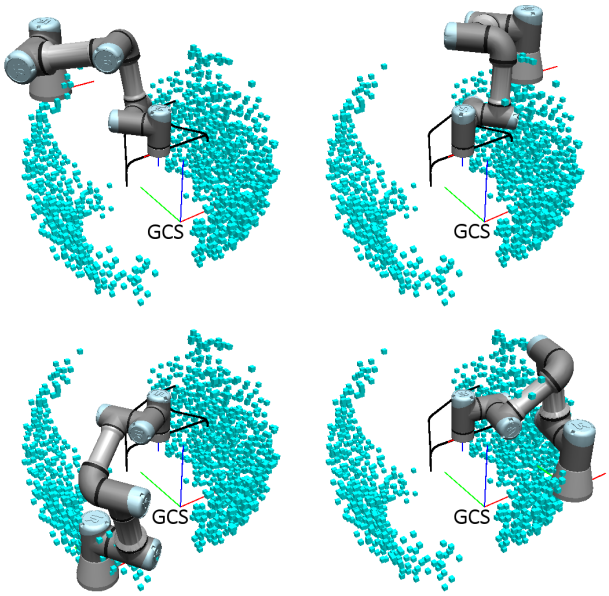


Figure 4. Example of a grid of valid robot locations for a given trajectory (A1). Each blue point represents a possible position of the robot base center point; the robot is displayed in four distinct sample locations. GCS represents the global coordinate system.

3. Results

For each of the ten trajectories, the values of $w_0, w_1, \ldots, w_6$ (5), and then f_f (8) were calculated based on the CoppeliaSim simulation for each particular possible robot location in the grid. The important values are summarized in Table 2, particularly the lowest fitness function value ${f_f}^B$ in the best robot location, which should, thus, be the selected location for the robot, provided it is physically possible. The table also shows the percentage of improvement that the best location offers compared to the worst location,

$$imp_W^B = 100 \times \left(1 - \frac{{f_f}^B}{{f_f}^W}\right), \tag{9}$$

which theoretically represents the greatest possible improvement in robot wear reduction. If we consider a random robot location versus the best one, the average improvement can be calculated compared to the average value

$$imp_A^B = 100 \times \left(1 - \frac{{f_f}^B}{{f_f}^A}\right). \tag{10}$$

Figure 5 shows the distribution of f_f values in all possible robot locations for each trajectory in the form of a standard box-plot diagram. The height of each shaded rectangle represents the interquartile range (third quartile minus first quartile), which indicates that 50% of all f_f values lie inside the rectangle. The small circles in each column represent the outliers—the topmost one corresponds to the worst location with ${f_f}^W$.

Arrangements of the robot locations in the 3D space around the corresponding trajectories are displayed in Figure 6—the color coding indicates the f_f values using the gradient red–yellow–green, where green is the best location (${f_f}^B$), and red is the worst (${f_f}^W$). This image also shows the positions of the best ("B"), worst ("W"), and three other robot locations ("1", "2", and "3") that will be used later in the experiments on a real robot. Note that the small cubes rendered in Figure 6 as a visual representation of the possible robot locations are shifted away from the ideal grid positions by a small random offset (less than half the grid spacing). This is to achieve better visual clarity by preventing moiré patterns and reducing the concealment of more distant cubes.

Table 2. Results for the 10 testing trajectories showing the number of valid robot locations n, the fitness function value in the best location ${f_f}^B$, in the worst location ${f_f}^W$, the average value ${f_f}^A$; and improvement of the best location against the worst imp_W^B (9) and the average imp_A^B location (10).

Traj.	n	${f_f}^B$	${f_f}^W$	${f_f}^A$	imp_W^B	imp_A^B
A1	800	0.0730	0.2053	0.1239	64.4%	41.1%
A2	564	0.0773	0.1577	0.1046	51.0%	26.1%
B1	2265	0.0217	0.1090	0.0461	80.1%	52.9%
B2	2236	0.0210	0.1067	0.0448	80.3%	53.1%
C1	2365	0.0769	0.1998	0.1239	61.5%	37.9%
C2	2214	0.0757	0.1828	0.1170	58.6%	35.2%
D1	866	0.0907	0.1994	0.1163	54.5%	22.0%
D2	828	0.0841	0.1829	0.1097	54.0%	23.3%
E1	2224	0.0279	0.1028	0.0532	72.8%	47.4%
E2	2151	0.0247	0.0984	0.0502	74.9%	50.7%

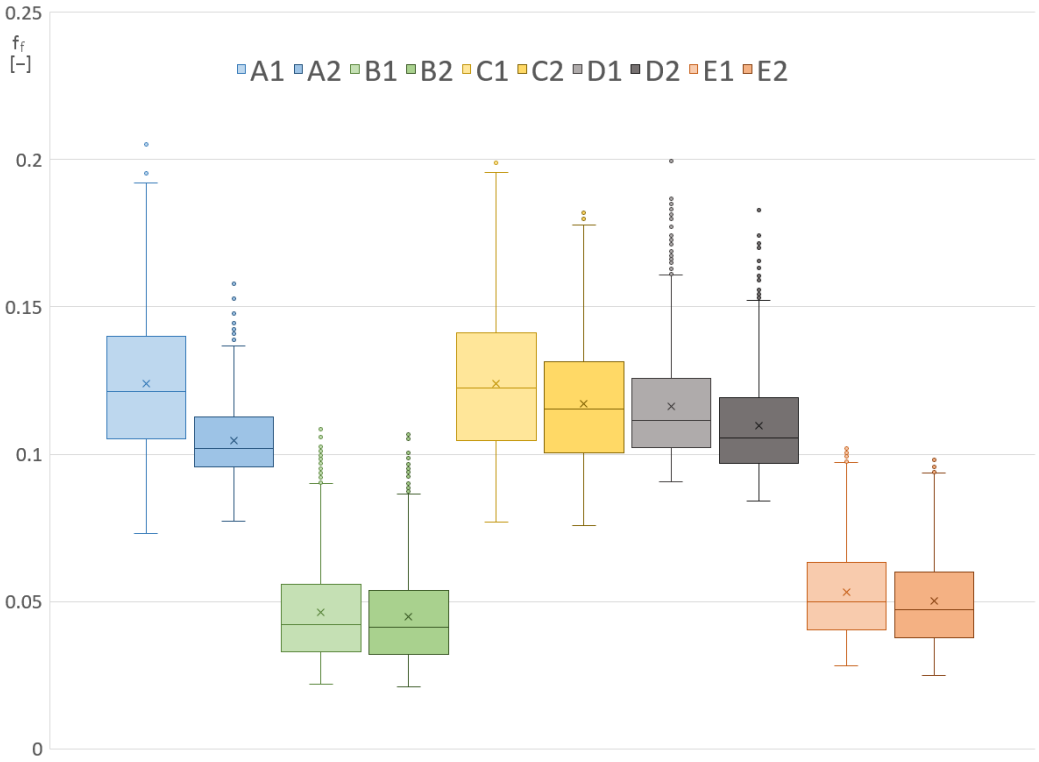
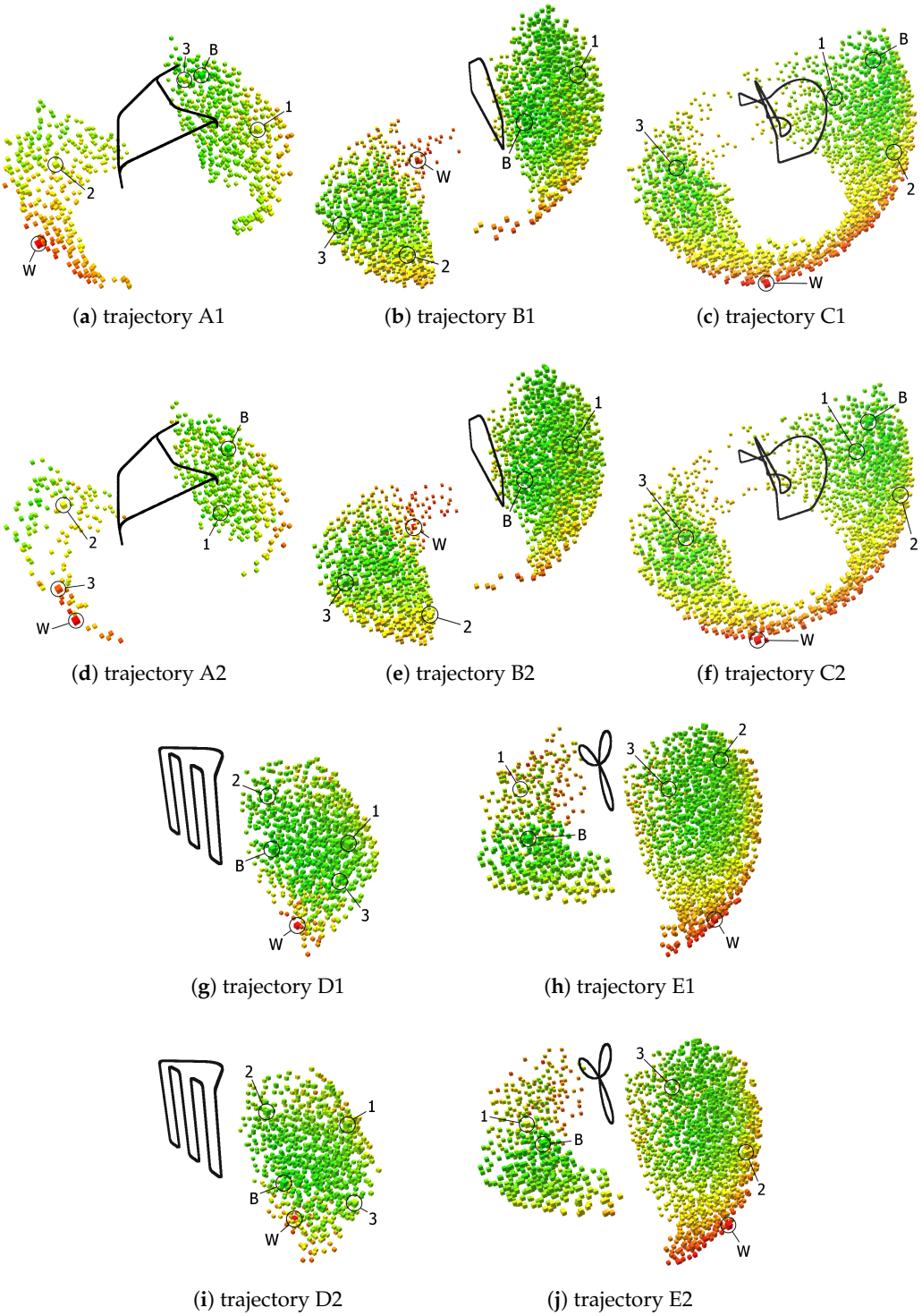


Figure 5. Statistical distribution of the f_f values in all possible robot locations for the ten testing trajectories (standard box-plot diagram—each box is bounded by the first and third quartile, the horizontal line represents the median, the × represents the arithmetic mean, and circles represent outliers).

To better demonstrate the meaning of the proposed optimization criterion, Figure 7 shows, for the two trajectories A1 and B1, the individual values of the relative joint wear factors $w_1, w_2, \ldots, w_6$ (5) for the best (${w_j}^B$) and worst (${w_j}^W$) robot location, together with the corresponding total fitness function value f_f (8). In the first example (trajectory A1), it is clear that, in the worst location, the third joint was extremely stressed and the six ${w_i}^W$ values differ quite considerably. In the best location, joints 2, 3, and 6 have almost the same ${w_i}^B$ values and, although the relative wear factor of the 6th joint increased, the overall ${f_f}^B$ value was much lower.

A similar situation can be seen in the second image (trajectory B1). Here, the relative wear factor values in the best location are not so well balanced–the overall fitness function value was reduced considerably nonetheless because the average value A (Equation (6)) lowered.

The reason for the big improvement in the third joint relative wear factor w_3 for both these trajectories can be seen in Figure 8—the green and red solid lines show the immediate power values for the best and worst robot location, respectively. The difference in the magnitude is evident.



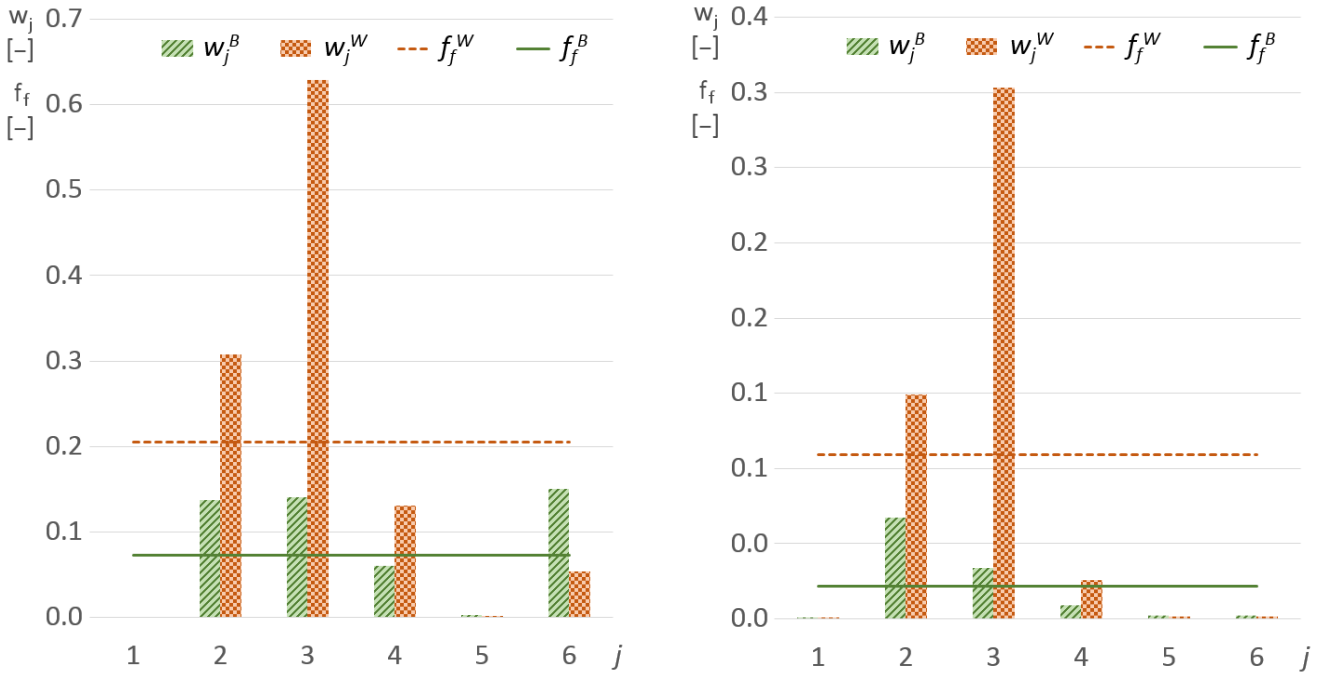
(**a**) trajectory A1 (**b**) trajectory B1 (**c**) trajectory C1

(**d**) trajectory A2 (**e**) trajectory B2 (**f**) trajectory C2

(**g**) trajectory D1 (**h**) trajectory E1

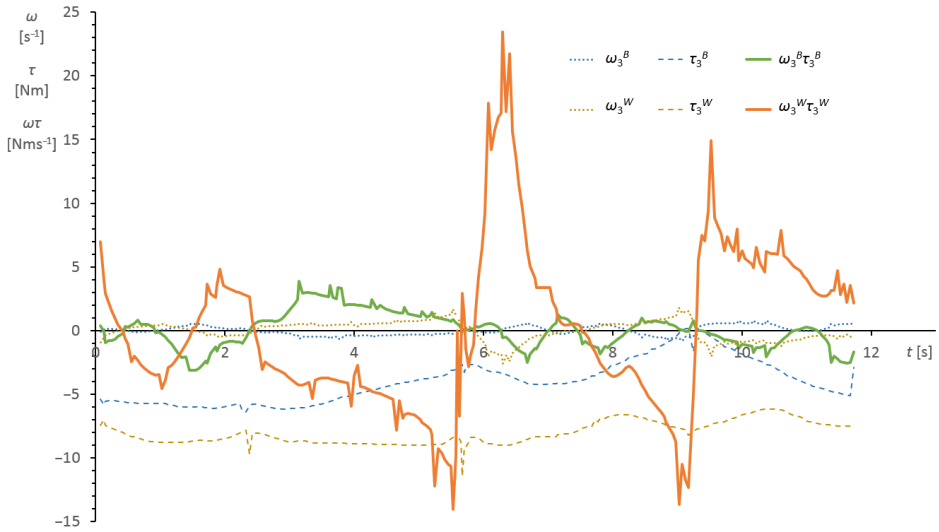
(**i**) trajectory D2 (**j**) trajectory E2

Figure 6. All valid robot locations in the grid; the color-coding indicates the f_f values using the gradient red–yellow–green, where green is the best location (B) and red is the worst (W). Locations numbered 1, 2, and 3 are the three other locations used in the experiment with a real robot.



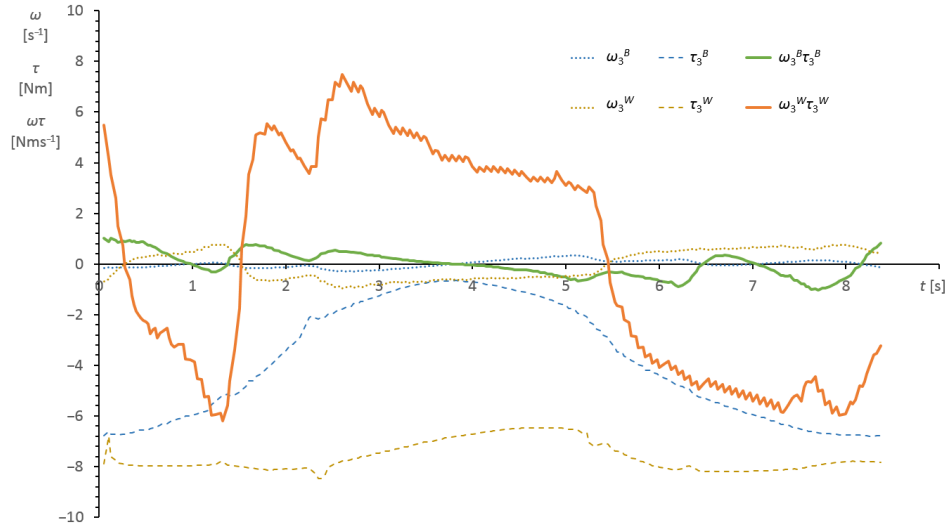
(**a**) trajectory A1 (**b**) trajectory B1

Figure 7. Detailed comparison of the components forming the fitness function value in the best (B, green) and worst (W, red) robot locations for two selected trajectories; displayed are the relative wear factors of each joint (w_j) and the total fitness function value (f_f).



(**a**) trajectory A1

Figure 8. *Cont.*



(**b**) trajectory B1

Figure 8. Comparison of the angular velocity ω_3, torque τ_3, and immediate power $\omega_3\tau_3$ of the third joint during the whole trajectories A1 and B1 for the best (^{B}) and worst (^{W}) robot locations.

4. Verification on a Real Robot

To experimentally verify the impact of the chosen fitness function on the real wear of the robot joints, it would be necessary to perform simultaneous long-term testing on multiple robots and then analyze the mechanical degradation of important components of the joint construction. This type of experiment was not feasible for us at this moment. Instead, we used experiments on a real robot to verify the accuracy of dynamical simulation in CoppeliaSim.

The real experiments were performed on the same robot as was previously used for the simulations—the Universal Robots UR3 collaborative robot. The robotic arm was mounted on a table (Figure 2), and, instead of changing the robot base location relative to the trajectory, the trajectory was appropriately shifted in relation to the robot. There were no tools nor other equipment mounted to the output interface flange, which corresponds to the simulation model described above.

The robot controller (CB3 controller, firmware version 3.10.0) provides state messages via the RTDE (Real-Time Data Exchange) protocol based on TCP/IP communication. This protocol allows reading the actual state of the robot with a frequency of 125 Hz. The RTDE messages can be configured and include the actual and target values of kinematic parameters, such as the position, velocity, and acceleration, as well as the currents of individual joints, the overall current of the whole robot, and the target torque values—which are needed for our experiment.

The experiment was executed for the optimal (best) robot location found by the simulation, for the worst robot location, and for three other locations that were manually selected from the set of possible locations to cover various sections of the whole space. The locations are hereafter referred to as "B" (best), "W" (worst), and "1", "2", and "3" (the other locations). The three numbered locations are sorted from the lowest to the highest f_f values according to the simulation results (a lower f_f value indicates a better robot location). For each trajectory and robot location, the robot was programmed to go through the whole trajectory five times in a row. During this time, the integral (sum) was calculated according to (4), and the final value was then divided by five.

All tested real robot locations are depicted in Figure 6. Numerical values of the f_f values in all five locations for all ten trajectories are listed in Tables 3 and 4. Table 3 shows the results from the simulation, the values here are sorted from lowest to highest ("B", "1", "2", "3", and "W"). Table 4 shows the values from real experiments—as can be seen, the

best locations were identical; the worst locations were also mostly identical, except for trajectory D2.

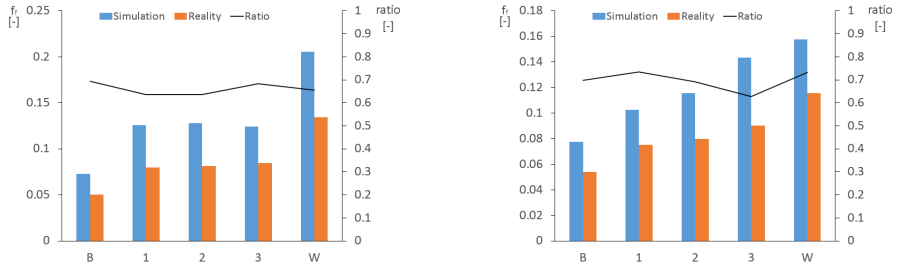
Table 3. Fitness function values in the five robot locations for individual trajectories—results from the simulation. Color gradient red–yellow–green is used for better visual representation of the value (green is the best, red is the worst).

	B	1	2	3	W
A1	0.0730	0.1256	0.1281	0.1241	0.2054
A2	0.0774	0.1025	0.1156	0.1434	0.1578
B1	0.0217	0.0357	0.0389	0.0484	0.1090
B2	0.021	0.0289	0.0310	0.0588	0.1068
C1	0.0770	0.1074	0.1195	0.1421	0.1998
C2	0.0758	0.0939	0.1040	0.1160	0.1828
D1	0.0907	0.1045	0.1077	0.1187	0.1995
D2	0.0841	0.0967	0.1157	0.1203	0.1830
E1	0.028	0.0358	0.0388	0.0642	0.1028
E2	0.0247	0.0371	0.0585	0.0606	0.0985

Table 4. Fitness function values in the five robot locations for individual trajectories–results from the real robot. Color gradient red–yellow–green is used for better visual representation of the value (green is the best, red is the worst).

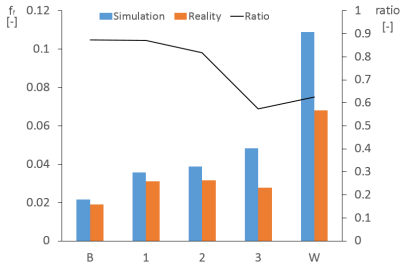
	B	1	2	3	W
A1	0.0506	0.0799	0.0815	0.0848	0.1343
A2	0.054	0.0752	0.0799	0.0900	0.1155
B1	0.019	0.0312	0.0317	0.0278	0.0681
B2	0.0191	0.0263	0.0280	0.0348	0.0679
C1	0.0635	0.0760	0.0828	0.0874	0.1176
C2	0.0621	0.0625	0.0810	0.0808	0.1124
D1	0.078	0.0877	0.0940	0.0783	0.1015
D2	0.0777	0.0869	0.1038	0.0857	0.0958
E1	0.0239	0.0310	0.0348	0.0457	0.0599
E2	0.025	0.0355	0.0481	0.0446	0.0621

Table 5 shows the ratios between the real and simulated values; ideally, these values should all be equal to 1. In general, it can be stated that the f_f values acquired by the real experiments were lower than the values from the simulation, and the average ratio was $r = 0.745$ with the standard deviation equal to $\sigma_r = 0.126$. Figure 9 displays a comparison of the simulated and real values in the form of a bar graph, together with the ratio r. In this image, it can be observed that, for the same trajectory, the ratio r typically lowers (deteriorates) with increasing the f_f values. This will be further discussed in the Discussion section.

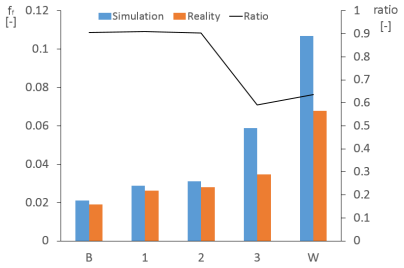


(**a**) trajectory A1

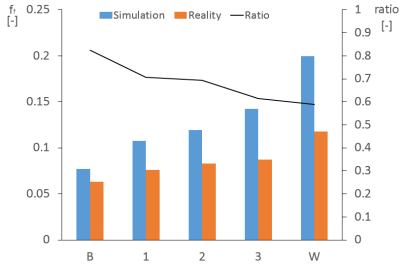
(**b**) trajectory A2



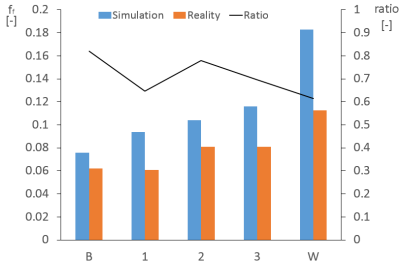
(**c**) trajectory B1



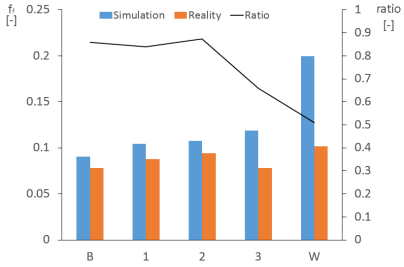
(**d**) trajectory B2



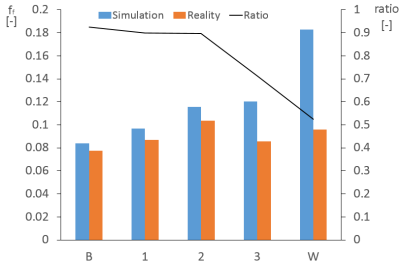
(**e**) trajectory C1



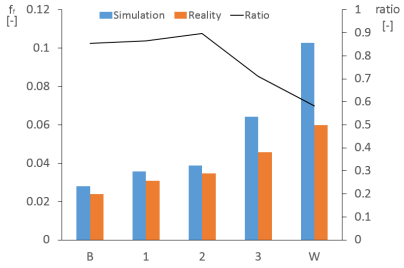
(**f**) trajectory C2



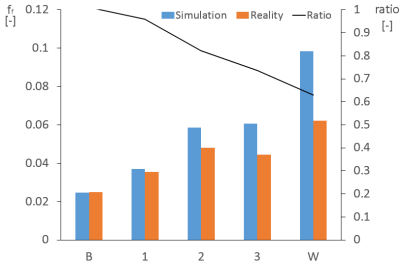
(**g**) trajectory D1



(**h**) trajectory D1



(**i**) trajectory E1



(**j**) trajectory E2

Figure 9. Comparison of the fitness function values acquired from simulation and experiments on the real robot; the black lines represent the ratio between the values.

Table 5. Ratios between the real and simulated values of fitness function in the five robot locations for individual trajectories. Color gradient red–white is used for better visual representation of the value (white is the ideal ratio of 1.0, red is the ratio 0.5).

	B	1	2	3	W
A1	0.6932	0.6359	0.6362	0.6833	0.6541
A2	0.6985	0.7334	0.6912	0.6278	0.732
B1	0.8743	0.8722	0.8168	0.5735	0.6244
B2	0.9056	0.9096	0.9025	0.5915	0.6359
C1	0.8246	0.7075	0.6933	0.6151	0.5887
C2	0.8199	0.6657	0.7792	0.6965	0.6149
D1	0.8594	0.8393	0.8728	0.6591	0.5087
D2	0.9243	0.8984	0.8968	0.7126	0.5238
E1	0.8541	0.8658	0.8960	0.7118	0.5825
E2	1.0100	0.9590	0.8224	0.7360	0.6303

5. Discussion

The paper describes the grid approach to finding the optimal robot location, where all possible robot locations are evaluated using a simulation in CoppeliaSim, and then the location with the best (lowest) fitness function f_f value is selected. In our case, there was an additional preprocessing step that found all valid robot locations using a simple and fast custom-made simulation system to increase the speed of the process. This step is not required, and therefore the research can be reproduced without this custom simulation system.

The grid approach can alternatively be replaced by using some optimization algorithms, for example, the Particle Swarm Optimization (PSO) [31], which could further reduce the time needed to find the optimal solution. However, PSO would not provide the same type of comprehensive analysis of the whole space around the trajectory, and could also possibly return a local minimum instead of the global one.

The method was demonstrated and tested on ten sample trajectories and the collaborative robot UR3. As can be seen from the results in Table 2, the percentage improvement in the fitness function between the worst and the best possible robot location ranged from 54% to 80.3%. This is mostly a theoretical improvement, as the worst-rated locations would likely be not chosen by the system integrator or workplace project architect for other reasons (they are typically on the edge of the working area of the robot or close to a singular configuration). If we instead compare the best robot location to the average f_f value of all possible locations, the improvement still ranges from 22% to 53.1%; therefore, it is clear that some interesting reduction in the robot joints wear can be achieved by selecting the optimal robot location instead of a "random" one.

This method can be used in practice, provided there is a dynamic model of the selected robot available for some suitable simulation system (for example, CoppeliaSim). It is necessary to properly define the whole simulation model of the workplace, including any potential obstacles, otherwise, the returned optimal robot location could be invalid due to a collision. Nonetheless, it is still important to verify that the optimal location is feasible from the point of view of energy connections and other similar restrictions that cannot be included in the simulation.

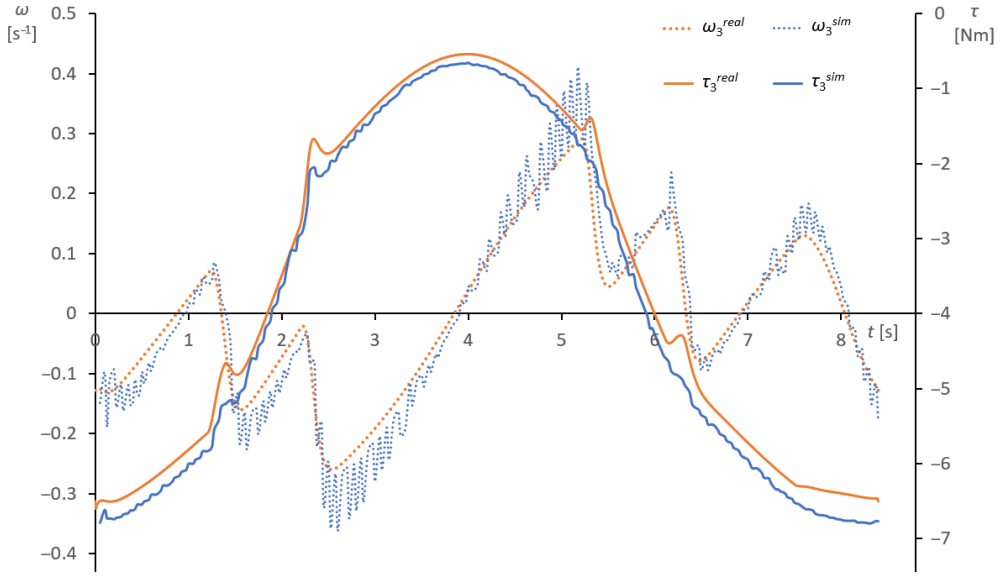
The dynamic model should include all properties and phenomena that noticeably affect the torque values in joints of the robot, as the torques are one of the main inputs for the proposed method. It is, thus, necessary to properly define and simulate also the payload in the end-effector, including any technological forces caused by the effector during, for example, cutting, milling, and spraying. The simulation system CoppeliaSim chosen in our demonstration is capable of simulating all these effects.

Experiments on a real robot UR3 verified that the simulation model in CoppeliaSim matched the reality acceptably well. The absolute values of f_f acquired by the simulation and from the real robot differed approximately by the scale factor of 0.745 ± 0.126.

The difference between simulation and reality was caused especially by the simulation model of the UR3 robot, which is likely not absolutely perfect as far as mass properties are concerned, and also by the dynamic simulation engine, which performed some simplifications. More important is that, in the relative evaluation of individual robot locations, the real experiments led to very similar results as the simulation—as can be seen in Tables 3–5. There were some distinct deviations only in one path of robot end-point movement (trajectories D1, D2). It can be, thus, stated that, to improve the robot joints wear, potentially expensive and time-consuming real-world experiments are not necessary; a simulation suffices.

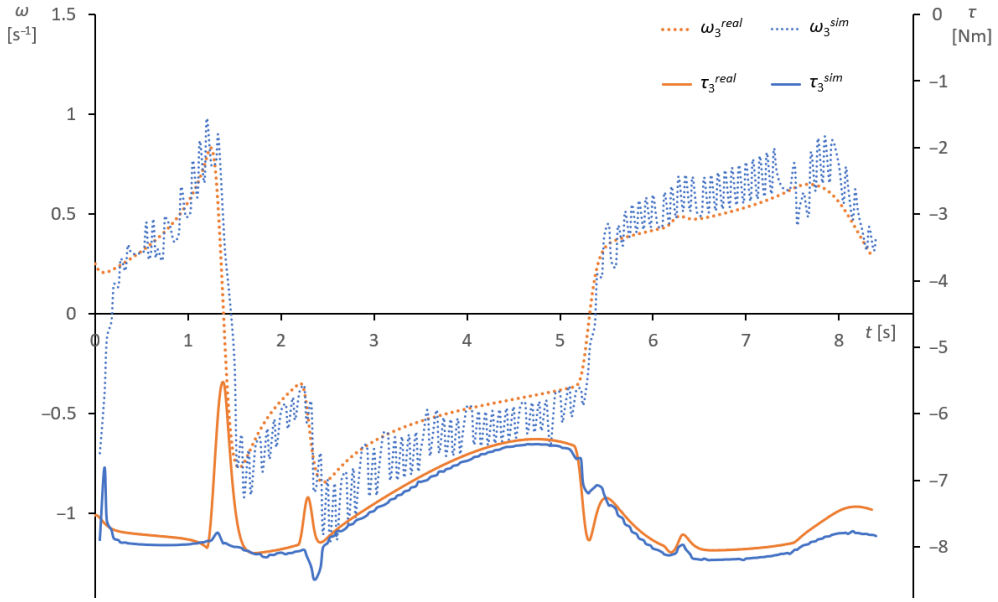
To demonstrate and analyze the source of the difference between the values from the simulation and from the real robot, Figure 10 shows an example of the simulated torque τ_3^{sim} and velocity ω_3^{sim} values directly compared with the values from the real robot (τ_3^{real}, ω_3^{real}), for the third joint of the robot during the B1 trajectory. The absolute value of τ_3^{real} was generally lower than the absolute value of τ_3^{sim}, which was likely caused by inaccurate mass parameters of the robot links in the simulation model. This is the reason why the f_f values from the real robot were mostly lower than the values from the simulation (see Table 5). The noise in the ω_3^{sim} values was caused by the discrete character of the simulation.

Figure 10b shows that, for the worst robot location, the τ_3^{real} values differed significantly from τ_3^{sim} in several peaks that correspond to moments with noticeable acceleration in the movement. This is probably caused by some simplifications in the dynamics engine in the simulation system or by some advanced algorithms in the control system of the real robot. Generally, in worse robot locations (with higher f_f values), the robot joints must perform movements with higher acceleration, which causes this additional source of difference between the simulation and reality.



(**a**) trajectory B1, the best robot location

Figure 10. *Cont.*



(**b**) trajectory B1, the worst robot location

Figure 10. Comparison of the real and simulated values of angular velocity ω_3 and torque τ_3 of the third joint during the whole trajectory B1. (**a**) Robot placed in the best location. (**b**) Robot placed in the worst location.

The proposed optimization process of a robotized workplace can be applied in the design phase when it is easy to make modifications to the workplace layout. Another option is to use the optimization for an existing workplace when other types of optimization are not possible (the trajectory is fixed, etc.).

6. Conclusions

The goal of this paper was to present a methodology for the optimization of a robot manipulator base position relative to the given trajectory of movement of the manipulator end-point. The optimization algorithm was based on the principle of minimization of the chosen fitness function, which comprises the arithmetic mean and standard deviation of the relative wear factors of all robot joints (8). The goal was to balance and minimize the wear of the drive chain in the robot joints, where the wear is approximated as the amount of mechanical work done by the joint, while also taking into account the abilities of the particular joint (the maximal permissible torque and velocity).

Future work will include verification on different types of robots and with various payloads. To verify the real impact on the wear of robot joints and, thus, the reduction of the robot lifetime, a long-term experiment would have to be performed simultaneously on at least two identical robots. One robot would be placed in a location with a good (low) f_f value and the other—reference—robots would perform the same task from locations with higher f_f values. Afterward, the robots would have to be partially disassembled to analyze and compare the mechanical degradation of the joints.

The cases when the workplace is used alternately for several different tasks and, thus, the robot performs more than one movement trajectory could also be addressed. The trajectories could either be combined into one for the simulation, or the method could be applied to each trajectory separately and then the results would be combined, for example by interpolation using weight coefficients given by the frequency of use of the trajectories.

Although this method does not require extremely precise dynamic simulation, this aspect could also be improved by creating a more accurate dynamic model of the robot, for example using some of the dynamic parameter identification method [32]. However, the simulation of friction and other complex phenomena will always be simplified in commonly available simulation systems.

Author Contributions: Conceptualization, T.K., Z.B. and V.K.; methodology, T.K., Z.B. and A.V.; software, T.K., A.V. and J.Š.; validation, T.K., A.V. and Z.B.; investigation, T.K., Z.B., V.K. and R.R.; writing—original draft preparation, T.K., Z.B. and V.K.; writing—review and editing, Z.B., A.V., V.K., J.Š. and R.R.; visualization, T.K.; supervision, Z.B.; project administration, Z.B. and V.K. All authors have read and agreed to the published version of the manuscript.

Funding: This article has been elaborated under support of the project Research Centre of Advanced Mechatronic Systems, reg. No. CZ.02.1.01/0.0/0.0/16_019/0000867 in the frame of the Operational Program Research, Development and Education.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data presented in this study are available on request from the corresponding author. The data are not publicly available due to funding project restrictions.

Conflicts of Interest: The authors declare no conflict of interest. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript, or in the decision to publish the results.

Abbreviations

The following abbreviations are used in this manuscript:

API	Application programming interface
CSO	Chicken swarm optimization
GCS	Global coordinate system
IK	Inverse kinematics
PSO	Particle swarm optimization
RTDE	Real-time data exchange
TCP/IP	Transmission control protocol/Internet protocol
UR3	Universal Robots 3

References

1. Diaz, J.R.; Mukherjee, P.; Verl, A. Automatic Close-optimal Workpiece Positioning for Robotic Manufacturing. *Procedia CIRP* **2018**, *72*, 277–284. [CrossRef]
2. Tangpattanakul, P.; Artrit, P. Minimum-time trajectory of robot manipulator using Harmony Search algorithm. In Proceedings of the 2009 6th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology, Chonburi, Thailand, 6–9 May 2009; Volume 1, pp. 354–357. [CrossRef]
3. Valente, A.; Baraldo, S.; Carpanzano, E. Smooth trajectory generation for industrial robots performing high precision assembly processes. *CIRP Ann.* **2017**, *66*, 17–20. [CrossRef]
4. Llopis-Albert, C.; Rubio, F.; Valero, F. Modelling an Industrial Robot and Its Impact on Productivity. *Mathematics* **2021**, *9*, 769. [CrossRef]
5. Piazzi, A.; Visioli, A. Global minimum-jerk trajectory planning of robot manipulators. *IEEE Trans. Ind. Electron.* **2000**, *47*, 140–149. [CrossRef]
6. Gasparetto, A.; Zanotto, V. A technique for time-jerk optimal planning of robot trajectories. *Robot. Comput. Integr. Manuf.* **2008**, *24*, 415–426. [CrossRef]
7. Hu, S.; Kang, H.; Tang, H.; Cui, Z.; Liu, Z.; Ouyang, P. Trajectory Optimization Algorithm for a 4-DOF Redundant Parallel Robot Based on 12-Phase Sine Jerk Motion Profile. *Actuators* **2021**, *10*, 80. [CrossRef]
8. Mu, Y.; Zhang, L.; Chen, X.; Gao, X. Optimal Trajectory Planning for Robotic Manipulators Using Chicken Swarm Optimization. In Proceedings of the 2016 8th International Conference on Intelligent Human-Machine Systems and Cybernetics (IHMSC), Hangzhou, China, 27–28 August 2016; Volume 2, pp. 369–373. [CrossRef]
9. Lim, Z.Y.; Ponnambalam, S.G.; Izui, K. Nature inspired algorithms to optimize robot workcell layouts. *Appl. Soft Comput.* **2016**, *49*, 570–589. [CrossRef]
10. Bukata, L.; Šůcha, P.; Hanzálek, Z.; Burget, P. Energy Optimization of Robotic Cells. *IEEE Trans. Ind. Inform.* **2017**, *13*, 92–102. [CrossRef]
11. Gadaleta, M.; Berselli, G.; Pellicciari, M. Energy-optimal layout design of robotic work cells: Potential assessment on an industrial case study. *Robot. Comput. Integr. Manuf.* **2017**, *47*, 102–111. [CrossRef]
12. Pellicciari, M.; Berselli, G.; Leali, F.; Vergnano, A. A method for reducing the energy consumption of pick-and-place industrial robots. *Mechatronics* **2013**, *23*, 326–334. [CrossRef]

13. Meike, D.; Pellicciari, M.; Berselli, G. Energy Efficient Use of Multirobot Production Lines in the Automotive Industry: Detailed System Modeling and Optimization. *IEEE Trans. Autom. Sci. Eng.* **2014**, *11*, 798–809. [CrossRef]
14. Paes, K.; Dewulf, W.; Elst, K.V.; Kellens, K.; Slaets, P. Energy Efficient Trajectories for an Industrial ABB Robot. *Procedia CIRP* **2014**, *15*, 105–110. [CrossRef]
15. Spensieri, D.; Carlson, J.S.; Bohlin, R.; Kressin, J.; Shi, J. Optimal Robot Placement for Tasks Execution. *Procedia CIRP* **2016**, *44*, 395–400. [CrossRef]
16. Biesinger, F.; Meike, D.; Krass, B.; Weyrich, M. A Case Study for a Digital Twin of Body-in-White Production Systems General Concept for Automated Updating of Planning Projects in the Digital Factory. In Proceedings of the 2018 IEEE 23rd International Conference on Emerging Technologies and Factory Automation (ETFA), Turin, Italy, 4–7 September 2018; Volume 1, pp. 19–26. [CrossRef]
17. Miková, Ľ.; Kelemen, M.; Virgala, I.; Michna, M. Simulation Model of Manipulator for Model Based Design. *Appl. Mech. Mater.* **2014**, *611*, 175–182. [CrossRef]
18. Georgoulas, G.; Loutas, T.; Stylios, C.D.; Kostopoulos, V. Bearing fault detection based on hybrid ensemble detector and empirical mode decomposition. *Mech. Syst. Signal Process.* **2013**, *41*, 510–525. [CrossRef]
19. Qian, H.M.; Li, Y.F.; Huang, H.Z. Time-variant reliability analysis for industrial robot RV reducer under multiple failure modes using Kriging model. *Reliab. Eng. Syst. Saf.* **2020**, *199*, 106936. [CrossRef]
20. Guangjian, W.; Lin, C.; Li, Y.; Shuaidong, Z. Research on the dynamic transmission error of a spur gear pair with eccentricities by finite element method. *Mech. Mach. Theory* **2017**, *109*, 1–13. [CrossRef]
21. Cubillo, A.; Perinpanayagam, S.; Esperon-Miguez, M. A review of physics-based models in prognostics: Application to gears and bearings of rotating machinery. *Adv. Mech. Eng.* **2016**, *8*, 1687814016664660. [CrossRef]
22. Zhang, X.; Jiang, G.; Zhang, H.; Yun, X.; Mei, X. Time-dependent reliability analysis of harmonic drive based on transient FEA and accelerated life test. *Eng. Comput.* **2020**. [CrossRef]
23. Bian, L.; Gebraeel, N.; Kharoufeh, J.P. Degradation modeling for real-time estimation of residual lifetimes in dynamic environments. *IIE Trans.* **2015**, *47*, 471–486. [CrossRef]
24. Koike, H.; Itakura, K.; Okazaki, S.; Takamiya, M.; Kanemasu, K.; Kida, K. Measurement of Backlash and Fatigue Wear of PEEK Bush in Robot Joint under Middle Load. In *Applied Mechanics and Materials*; Trans Tech Publications Ltd.: Stafa-Zurich, Switzerland, 2013; Volume 418, pp. 38–43. [CrossRef]
25. Bittencourt, A.C.; Axelsson, P.; Jung, Y.; Brogårdh, T. Modeling and Identification of Wear in a Robot Joint under Temperature Uncertainties. *IFAC Proc. Vol.* **2011**, *44*, 10293–10299. [CrossRef]
26. Lugt, P.M. A Review on Grease Lubrication in Rolling Bearings. *Tribol. Trans.* **2009**, *52*, 470–480. [CrossRef]
27. Aivaliotis, P.; Arkouli, Z.; Georgoulias, K.; Makris, S. Degradation curves integration in physics-based models: Towards the predictive maintenance of industrial robots. *Robot. Comput. Integr. Manuf.* **2021**, *71*, 102177. [CrossRef]
28. Peternel, L.; Tsagarakis, N.; Ajoudani, A. A Method for Robot Motor Fatigue Management in Physical Interaction and Human-Robot Collaboration Tasks. In Proceedings of the 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Madrid, Spain, 1–5 October 2018; pp. 2850–2856. [CrossRef]
29. Rodrigues, L.R. Remaining Useful Life Prediction for Multiple-Component Systems Based on a System-Level Performance Indicator. *IEEE/ASME Trans. Mechatron.* **2018**, *23*, 141–150. [CrossRef]
30. Jacobs, S.; Rios-Gutierrez, F. Self organizing maps for monitoring parameter deterioration of DC and AC motors. In Proceedings of the 2013 Proceedings of IEEE Southeastcon, Jacksonville, FL, USA, 4–7 April 2013; pp. 1–6. [CrossRef]
31. Poli, R.; Kennedy, J.; Blackwell, T. Particle swarm optimization. *Swarm Intell.* **2007**, *1*, 33–57. [CrossRef]
32. Kovincic, N.; Mueller, A.; Gattringer, H.; Weyrer, M.; Schlotzhauer, A.; Brandstötter, M. Dynamic parameter identification of the Universal Robots UR5. In Proceedings of the Austrian Robotics Workshop 2019, Steyr, Austria, 9–10 May 2019. [CrossRef]