Optimizing Wrist Placement of a Combined Hand-Arm System for Maximum Manipulability

Author: Jaime Aparicio Estrems

Supervisor: Adrian Sieler

***Abstract*— The optimal placement of a robot's end-effector is critical to achieving maximum manipulability in various tasks. In this paper, we present a method for optimizing the wrist placement of a combined hand-arm system with seven degrees of freedom to maximize the manipulability of the robot. The method is based on the Dividing Rectangles (DIRECT) algorithm, which is a deterministic global optimization algorithm that can find the global optimum of a continuous function. The objective function for the optimization problem is defined as the volume of the manipulability ellipsoid, which is a measure of the overall manipulability of the robot. We introduce joint limits and workspace constraints to ensure that the robot operates within its physical limitations and can perform the required tasks. We apply the DIRECT algorithm to find the optimal wrist placement of the robot that maximizes the manipulability volume while satisfying the constraints.**

I. Introduction

As the field of robotics continues to advance, there is an increasing need for efficient and effective robotic systems that can perform complex tasks in a variety of environments. One key factor in determining the performance of a robotic system is its manipulability, or the ability of the system to manipulate its environment. Manipulability is a critical factor in applications ranging from manufacturing to space exploration, where precise and dexterous manipulation is often required. By ensuring that the robotic system's kinematic chain is set in a configuration that maximizes its manipulability, we can provide this robotic system with a more human-like motion, in which not just the end-effector position and orientation, but all the system's kinematic chain position and orientation cooperate to perform the desired task.

The research presented in this paper aims to address the challenge of optimising the control of wrist placement of a robotic manipulator with optimal manipulability. To achieve this goal, we propose an approach that combines already-known mathematical modelling on manipulability with a heuristic optimization technique, the DIRECT algorithm, to analyze the manipulability of a 7 Degrees Of Freedom (DOF) hand-arm robotic system. The results obtained will be compared and benchmarked with a gradient-based approach optimizer, designed by a colleague with the same main goal. By developing a comprehensive understanding of manipulability, we aim to improve the performance of robotic manipulators in a wide range of applications.

This research is motivated by the growing demand for advanced robotic systems that can perform complex tasks with high efficiency and accuracy. By improving manipulability, we can enhance the performance of robotic systems in a variety of domains, from manufacturing to healthcare. The proposed approach has the potential to make significant contributions to the field of robotics, and we hope that it will inspire further research in this area.

II. Related Work

Manipulability analysis has been a subject of interest in robotics research for many years. Researchers like Tsuneo Yoshikawa [1] and Nikolaus Vahrenkamp [2] have contributed significantly to the field by developing mathematical models and algorithms for analyzing the manipulability of robotic systems. Yoshikawa [1] focuses on developing indices to quantify the manipulability of robotic mechanisms, while Vahrenkamp [2] research explores manipulability analysis in the context of redundant manipulators.

In addition to manipulability analysis, recent research has also focused on developing analytical inverse kinematics solutions for robotic manipulators. Yanhao He and Steven Liu [3] have proposed a geometrical solver for 7-DOF manipulators. Their work demonstrates the feasibility of achieving accurate and efficient inverse kinematics solutions for complex robotic manipulators. The Franka Emika Panda manipulator has also been the subject of research into dynamic identification and parameter retrieval. Claudio Gaz, et al. [4] explore the use of penalty-based optimization to identify the dynamic parameters of the Panda robot. Gaz's research demonstrates the potential of optimization techniques to improve the accuracy of robotic manipulators.

In the context of optimizing wrist placement of a combined hand-arm system for maximum manipulability, the work of these researchers is highly relevant. By analyzing the manipulability of a robotic system, we can identify optimal wrist placement and design parameters that maximize the system's performance. Additionally, the geometrical solver proposed by He and Liu [3] can be used to determine the optimal position and orientation of the wrist relative to the hand and arm.

Furthermore, the dynamic identification techniques proposed by Gaz [4] can be used to identify the system parameters required for accurate manipulability analysis. By accurately characterizing the dynamic properties of the hand-arm system, we can improve the accuracy of manipulability analysis and optimize the design parameters for maximum performance.

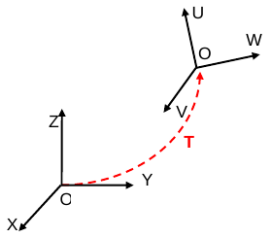

Fig. 1: Translation and rotation from frame $\mathbf{O}(\mathbf{x},\mathbf{y},\mathbf{z})$ to $\mathbf{O'}(\mathbf{v},\mathbf{u},\mathbf{w})$.

III. Technical Section

A. Mathematical background

In order to understand forward kinematics, it is important to first define some key terms. A Euclidean space is a mathematical space that is characterized by its distance metric and the number of dimensions it has. In the context of robotics, a Euclidean space is typically three-dimensional, although higher-dimensional spaces can also be used. The position and orientation of a robot's end effector in a Euclidean space are commonly referred to as its pose. In our case, we have a system defined by a 6-dimensional space, 3 dimensions for translation (x,y,z) and 3 dimensions for rotation (α,β,γ).

The forward kinematics problem involves determining the pose of a robot's end effector based on the values of its joint angles. This is done by applying a series of transformations that relate the position and orientation of each joint to the position and orientation of the end effector. These transformations can be represented using matrices, and the product of these matrices provides the forward kinematics solution. If we take $\mathbf{FK}(\mathbf{x})$ as the forward kinematics function of the $\mathbf{x}$ vector, it's result will be a pose vector $\mathbf{p}$. Which can be expressed as: $\mathbf{FK}(\mathbf{x}) = \mathbf{p}$.

Inverse kinematics is used to determine the joint angles required to position the end effector in a desired location and orientation. This involves solving a system of equations that relate the joint angles to the position and orientation of the end effector. If we take $\mathbf{IK}(\mathbf{p})$ as the inverse kinematics function of the pose vector $\mathbf{p}$, it's result will be a $\mathbf{x}$ vector. Which can be expressed as: $\mathbf{IK}(\mathbf{p}) = \mathbf{x}$.

To solve the forward kinematics problem, the homogeneous transformation matrix (HTM) method is the one used in this paper. It involves using a single transformation matrix to relate the position and orientation of the end effector to the position and orientation of the robot's base. In this matrix, for a rotation and translation from frame $\mathbf{O}(\mathbf{x},\mathbf{y},\mathbf{z})$ to frame $\mathbf{O'}(\mathbf{v},\mathbf{u},\mathbf{w})$, as seen in figure 1, we'll obtain a HTM $\mathbf{T}$ with a rotational component **rot** size (3x3) and a translational component **trans** size (3x1), as seen in equation 1.

$$ {}^{\mathbf{O}}_{\mathbf{O'}}\mathbf{T}_{\mathbf{(4x4)}} = \begin{bmatrix} \mathbf{rot}_{\mathbf{(3x3)}} & \mathbf{trans}_{\mathbf{(3x1)}} \\ \mathbf{0} & \mathbf{1} \end{bmatrix} \tag{1} $$

To solve the inverse kinematics problem, which involves determining the joint angles required to position the end effector in a desired location and orientation, we use the Jacobian matrix of the specific function. The Jacobian matrix is a matrix of partial derivatives that describes the position of the end effector as a function of the joint configuration, relating every dimension $(\mathbf{x},...,\psi)$ to every joint $(\mathbf{q1},...,\mathbf{q_n})$. In other words, the Jacobian matrix relates the configuration of the joints to the position of the end effector, as seen in figure 2.

$$ \mathbf{J}_a = \begin{bmatrix} \frac{\partial f_x}{\partial q_1} & \cdots & \frac{\partial f_x}{\partial q_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial f_\psi}{\partial q_1} & \cdots & \frac{\partial f_\psi}{\partial q_n} \end{bmatrix} $$

Fig. 2: Jacobian matrix.

In order to incorporate additional constraints into the inverse kinematics problem, we'll implement the null space constraint. It allows the robot to move in a way that is consistent with a secondary objective, specifically a wrist placement constraint in our case, even while it is attempting to reach a primary objective, obtaining maximum manipulability.

The null space constraint is implemented by modifying the Jacobian matrix. The modified Jacobian matrix is known as the pseudo-inverse or the Moore-Penrose inverse. The pseudo-inverse of the Jacobian matrix can be used to determine the joint configuration required to position the end effector in a certain location while also satisfying the null space constraint. This allows the robot to maintain a desired posture, while still achieving the primary objective. In our case, this implementation to the Jacobian matrix $\mathbf{J}$, constraining the set of vectors $\mathbf{x}$ is formalised as:

$$ Null(\mathbf{J}) = \{\mathbf{x} |\ \mathbf{Jx} = 0\} \tag{2} $$

This will be applied to the Jacobian as follows: if we want to block the movement in a desired dimension $\mathbf{x}$ from all the possible dimensions $(x,y,z,\alpha,\beta,\gamma)$, it will be expressed as:

$$ \delta\mathbf{x} = \mathbf{J}\cdot\delta\mathbf{q} \tag{3} $$

Where $\mathbf{q}$ represents the joint configuration. And to constraint the movement in this dimension:

$$ 0 = \mathbf{J}\cdot\delta\mathbf{q} \tag{4} $$

Therefore, to obtain a projection with a nullspace constraint affecting a specific joint configuration $\mathbf{q}$,the implementation would as follows:

$$ proj_{null}(q) = (I - J^{\dagger}J)\cdot\mathbf{q} \tag{5} $$

B. Manipulability

One way to visualize the concept of manipulability is to imagine a sphere surrounding the end effector of a robot. The size of this sphere represents the space that the robot

can reach, and the shape of the sphere represents the robot's ability to move in different directions. If the sphere is elongated in a particular direction, it means that the robot has a high degree of manipulability in that direction. If the sphere is round, it means that the robot has equal manipulability in all directions. Following Yosikawa's proposed quantitative measure of manipulability [1], which can be defined as a scalar value **w**, given by:

$$\mathbf{w} = \sqrt{det(JJ^T)} \tag{6}$$

C. Joint Limits

Vahrenkamp et al. introduction of joint limits penalization [2], refers to the maximum and minimum range of motion for each joint. When joint limits are taken into account, the manipulability ellipsoid (which describes the range of motion of the robot's end effector) becomes distorted and elongated in certain directions. This distortion can lead to decreased manipulability in certain directions. This joint limit penalization prevents the robot from exceeding its joint limits, which are represented as l_j^- for the lower joint limit and l_j^+ for the upper joint limit for a specific joint j.

$$\mathbf{P} = 1 - exp(-k \cdot \prod_{j=1}^{n} \frac{(q_j - l_j^-)(l_j^+ - q_j)}{(l_j^+ - l_j^-)^2}) \tag{7}$$

By applying this equation, we'll obtain a penalization of $P = 0$ that when $q_j = l_j^-$ or $q_j = l_j^+$. If the joint configuration q_j exceeds the limits, we'll have a penalization of $P < 0$, and when the joint configuration q_j is between the range of the joint limits $l_j^- < q_j < l_j^+$, the penalization will be $0 < P < 1$, depending on the value of the parameter k.

By multiplying our manipulability function (6) w with this joint limits penalization, we can see that the manipulability value will decrease when the joint configurations get closer to the joint limits, and it will be maximum when it stays at the centre of the joint limits. Formulating a manipulability measure **M** from the combination of w and P:

$$\mathbf{M} = \mathbf{w} \cdot \mathbf{P} \tag{8}$$

D. DIRECT algorithm

The optimization algorithm chosen for this implementation is the DIRECT (Dividing Rectangles) algorithm, which is a deterministic global optimization algorithm used to find the global minimum of a continuous function. It was first introduced by Jones, Perttunen, and Stuckman in 1993 [5]. It is a member of the class of global optimization algorithms that are based on interval analysis. The algorithm works by dividing the search domain into smaller and smaller rectangles until the global minimum is found or the desired accuracy is achieved. At each iteration of the algorithm, the rectangles are divided based on the density of function evaluations in each rectangle. The algorithm keeps track of the best solution found so far and uses it to adaptively adjust the search domain. The search domain is reduced in regions where the function has been evaluated and no better solution has been found, while it is expanded in regions where the function has not been evaluated or where better solutions may exist. The DIRECT algorithm has several advantages over other global optimization methods. For example, it does not require any information about the derivative of the objective function, and it is more efficient in finding the global minimum of multimodal and nonconvex functions.

IV. Heuristic Optimization

The optimizer implementation was done using the Differentiable Robot Model, SciPy libraries, simulate with RViz using ROS packages for a Franka Panda robotic hand-arm system, together with other libraries and packages that can be found in the technical documentation. The optimization process has been implemented as follows.

A. Random Initialization

First, a random initial configuration for the joints is generated. This configuration is enforced to stay within the joint limits. As the main scenario, a constraint for the palm facing upwards is implemented, as can be seen in figure 3.

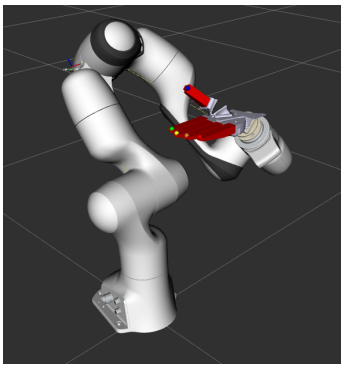

Fig. 3: Hand-arm robot simulation with random initial joint configuration.

B. Null space projection

The next step is to compute the projection with the null space constraint $proj_{null}(q)$, fixing the desired dimensions of our Jacobian matrix, such as:

$$proj_{null}(q) = (I - J^{\dagger}J) \cdot \mathbf{q} \tag{9}$$

C. DIRECT implementation

With the constraints already set, the DIRECT global optimizer algorithm is started. The algorithm will iterate recursively, searching for the optimal solution within the constrained displacements for $\delta\mathbf{q}$. At every iteration, it will compute the manipulability value **w** and the joint limit penalty **P**, returning the manipulability measure **M**. This process will repeat itself until a convergence condition is satisfied, or the maximum number of iterations is reached. The returned values are a pose matrix for the optimal solution reached, and the manipulability measure **M** for that joint configuration.

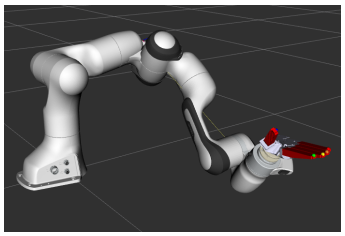

Fig. 4: Hand-arm robot simulation optimal result, after running the optimization algorithm.

V. Experimental Results

This section presents the achieved results and the configurations of DIRECT's hyperparameters used to obtain them. An initial optimisation round was done with the DIRECT algorithm, where no constraints were applied, in order to obtain the global optimum. The obtained global optimum for manipulability is $\mathbf{w} = 0.13167$.

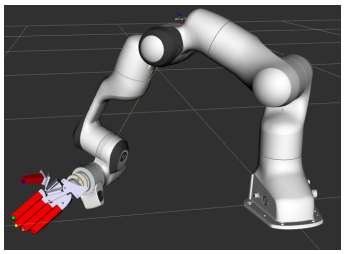

Fig. 5: Hand-arm robot simulation global optimal result, free optimization.

Figure 6 a scenario with all 3 rotational dimensions (α, β, γ) fixed, the incremental value of $\delta\mathbf{q}$ was set at 0.1, 0.05, 0.01 and 0.001, with 200 iterations for the DIRECT algorithm to converge.

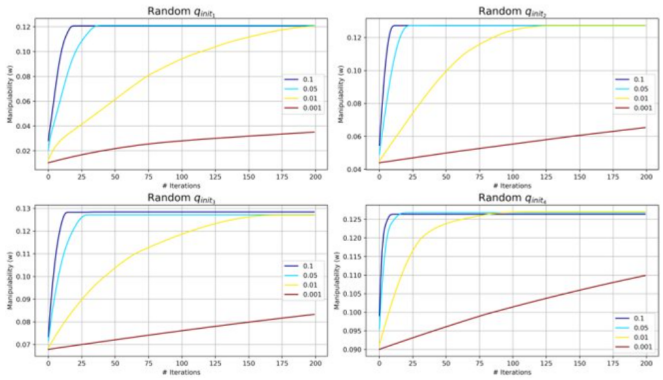

Fig. 6: $\delta\mathbf{q}$ increments, (α, β, γ) fixed.

Figure 7 scenario with all 3 translation dimensions (x, y, z) fixed, the incremental value of $\delta\mathbf{q}$ was set at 0.1, 0.05, 0.01 and 0.001, with 200 iterations for the DIRECT algorithm to converge.

Figure 8 a scenario with all 3 rotational dimensions (α, β, γ) fixed, the incremental value of $\delta\mathbf{q}$ was set at 0.1, with 200 iterations for the DIRECT algorithm to converge,

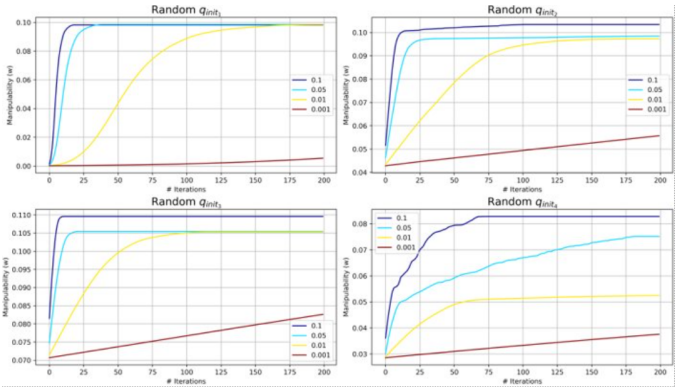

Fig. 7: $\delta\mathbf{q}$ increments, (x, y, z) fixed.

iterating the joint limits penalty $\mathbf{k}$ value between 100, 10000, 1000000 and 100000000.

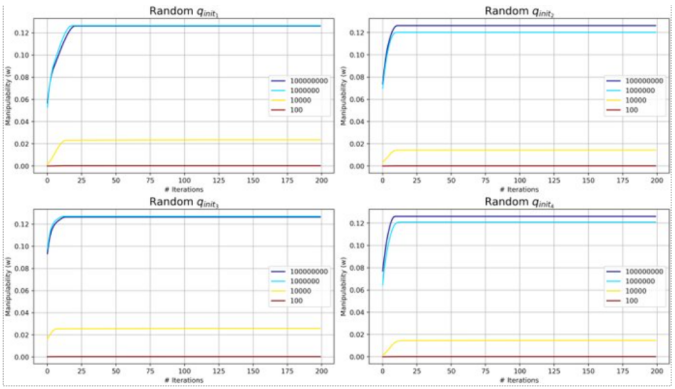

Fig. 8: $\mathbf{k}$ increments, (α, β, γ) fixed.

Figure 10 a scenario with all 3 rotational dimensions (x, y, z) fixed, the incremental value of $\delta\mathbf{q}$ was set at 0.1, with 200 iterations for the DIRECT algorithm to converge, iterating the joint limits penalty $\mathbf{k}$ value between 100, 10000, 1000000 and 100000000.

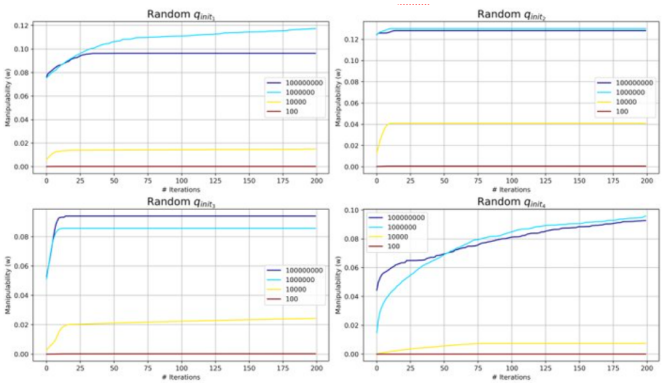

Fig. 9: $\mathbf{k}$ increments, (α, β, γ) fixed.

The results obtained from iterating the values from DIRECT's hyperparameters show that even though the manipulability optimization with null space projection for constraining the desired dimensions works, the limit joints penalty $\mathbf{k}$

factor has a strong influence. For lower **k** values, more influence of the joint limit penalty, it reaches lower manipulability values. In this scenario, maximum manipulability values are reached when the **k** value is so high that it strongly decreases the joint limit penalty. However, we can observe also by comparing figure 6 and figure 7, that the manipulability optimization is higher for scenarios with rotation fixed, compared to a scenario with translation fixed. These figures also show that for $\delta\mathbf{q}$ increments too small, the convergence to higher manipulability values requires more iterations.

VI. Benchmark

This section shows the comparison with Gradient-Based Optimization method, implemented by Clément Gillet. This benchmark has been run for 50 random initialized positions, with a pal facing upwards constraint, and implementing the null space projection fixing 2 rotation dimensions (α, β). The 3 scenarios were set with different joint limits penalty **k** values.

	Method	Average Time	Average Max Manipulability	Average Deviation (α, β)
$k = 5 \cdot 10^3$	DIRECT	5.2s	72%	(0.75, 0.3)
	ManGA	11.9s	56%	(0.006, 0.005)
$k = 10^4$	DIRECT	4.5s	71%	(0.7, 0.3)
	ManGA	7.4s	55%	(0.008, 0.007)
$k = 20 \cdot 10^3$	DIRECT	4.5s	74%	(0.68, 0.31)
	ManGA	7s	67%	(0.02, 0.02)

Fig. 10: Comparison for 3 different joint limits penalty **k** values, on speed, average manipulability and deviation from the constraint.

These results show that the Heuristic Optimization approach here presented has an average faster convergence time, reaching an average. But the difference in the average deviation (in radians) with respect to the constrained 2 rotation dimensions (α, β), is significant. The Gradient-Based Optimization method presents a much lower deviation.

VII. Conclusion

This paper has presented an approach to optimize the joint configuration to obtain the maximum manipulability for a hand-arm robotic system, using a heuristic global optimizer algorithm. The Joint Limit Penalty had a higher influence than expected, indeed a crucial factor for obtaining high manipulability values, demonstrating a practical implementation of Yoshikawa's [1] and Vahrenkamp's [2] work combined. Nevertheless, the method implementation was successful, with a satisfactory fast computation time.

Further investigation could dive into the deviation that this method presents, checking how this can affect the manipulability value, and trying to implement other heuristic global optimizer algorithms. An implementation for other constrained positions for the robot palm was started, but it was not fully successful, it could be worth completing it to adapt this method to different scenarios.

References

[1] T. Yoshikawa, "Manipulability of robotic mechanisms," *The International Journal of Robotics Research*, vol. 4, no. 2, pp. 3–9, 1985.

[2] N. Vahrenkamp, T. Asfour, G. Metta, G. Sandini, and R. Dillmann, "Manipulability analysis," 2012.

[3] Y. He and S. Liu, "Analytical inverse kinematics for franka emika panda – a geometrical solver for 7-dof manipulators with unconventional design," pp. 194–199, 11 2021.

[4] C. Gaz, M. Cognetti, A. Oliva, P. R. Giordano, and A. de Luca, "Dynamic identification of the franka emika panda robot with retrieval of feasible parameters using penalty-based optimization," *IEEE Robotics and Automation Letters, IEEE 2019, 4 (4)*, pp. 4147–4154.

[5] D. R. Jones, C. D. Perttunen, and B. E. Stuckman, "Lipschitzian optimization without the lipschitz constant," *Journal of Optimization Theory and Applications)*, vol. 79, pp. 157–181.