Poses Optimisation Methodology for High Redundancy Robotic Systems

Pedro Tavares[1,3], Pedro Costa[1,2], Germano Veiga[1,2], and António Paulo Moreira[1,2]

[1] FEUP - Faculty of Engineering of University of Porto, Portugal
[2] INESC TEC - INESC Technology and Science formerly INESC Porto Porto, Portugal
[3] SARKKIS robotics, Porto, Portugal

Abstract. The need for efficient automation methods has prompted the fast development in the field of Robotics. However, most robotic solutions found in industrial environments lack in both flexibility and adaptability to be applied to any generic task. A particular problem arises when robots are integrated in work cells with extra degrees of freedom, such as external axis or positioners. The specification/design of high redundancy systems, including robot selection, tool and fixture design, is a multi-variable problem with strong influence in the final performance of the work cell. This work builds on top of optimisation techniques to deal with the optimal poses reachability for high redundancy robotic systems. In this paper, it will be proposed a poses optimisation approach to be applicable within high redundancy robotic systems. The proposed methodology was validated by using real environment existent infrastructures, namely, the national CoopWeld project.

Keywords: Optimisation, Poses Determination, Cost Functions, Metaheuristics, High Redundancy Robotic Systems

1 Introduction

The robotic field's expansion leads to the desire of developing highly autonomous and efficient methodologies to address the prominent problems of not only research and development but also the needs pointed by the industrial corporations.

One of those problems in robotic systems is task management, that involves the integration of the motion planning in the work cell context, including technological process limitations, communication with external devices, automatic work cell calibration among others. Currently, there is no optimal tool to create an action sequence to complete a given task (currently handled by human experience). However, some studies point to the usage of optimisation functions to attend this problem [3, 1].

Recently, a wide range of applications have been developed towards the robotic implementation in harsh and / or repetitive tasks, such as welding, cutting and transportation, commonly found in the construction industries. These

tasks are fairly complex as they need to cope with several parameters. Robotic systems have to be autonomously capable of performing any coded task in a precise and efficient way. Thus, when developing systems that are going to be included into dangerous situations, the architecture that embraces the robotic system needs to consider robust elements. From the presented tasks, cutting and transportation seem to have a clear idea on where to improve and define a better and cleaner strategy while welding still needs to have some re-visitation on sensing, planning and adequate optimisation to any required task.

The specification of optimal design methodology of high redundancy systems, including robot selection, tools and fixtures design, is a multi-variable problem with strong influence in the final performance of the work cell. An active example of this problem is related to selection of flexible elements (such as the torching cables for welding operations) and their constraints.

In 2015, Graetz and Michaels, in their work entitled "Robots at work", analysed for the first time the economic impact of industrial robots pointing that robots significantly added increased value to industries [9]. Thus, this optimisation research topic applied to industrial work cells seems to be adequate to current state of industrial development.

Throughout this project, a novel design methodology capable of defining the correct work cell for a given operation in robotic applications is being developed in order to achieve industrial requirements and potentiate production goals. Furthermore some considerations on work cell motion and components reposition will assure an efficient function of the system.

The current paper is structured in five main section. Section 2 aims to provide an overview of the current state of optimisation technique appliance in research or industrial processes. Following, section 3, System Architecture and Heuristic-Based Approach, intends to present the defined optimisation structure, as well as the detail model description of a generic work cell submitted to the selected algorithms. Then, section 4, Optimisation Methodologies, will describe a set of algorithms to be applied in order to accurately find an optimal solution. Section 5, Validation, will provide preliminary results of the proposed optimisation approach and a comparison between selected algorithms. Section 6, Discussion and Future Perspectives, will summarise the contribution of this project to the scientific community and intended iterative of it.

2 Related Work

Robotic applications have become key to ensure process efficiency in a wide range of scientific fields. Several studies have suggested that there can be considerable gains in terms of profitability and reduced operation times on fields ranging from medical surgery to manufacturing proceedings [8, 12, 5].

Currently, optimisation methodologies have grown interest within research and industrial communities due to their success in solving multi-variable problems. Several algorithms have been defined and implemented showing great adaptability to complex tasks [17]. The main trend of such optimisation method-

ologies is related to heuristic-based algorithms. By defining detailed goals and constraints, such algorithm can iterative search for a ideal solution, and has been applied to several applications [15, 7].

Another key topic connected to optimisation and focusing in robotics is related to the work cell design, that comprises, among others, robot selection and fixture design. Cheng mentioned the simulation tool's advantages in order to develop a robotic work cell [4].

Furthermore, there have been authors claiming to find powerful enough methodologies to handle machining [2] and welding challenges [11, 10]. Still, despite its importance, existent robotic systems are focused on solving separate necessities while there is no optimal tool to properly design a generic robotic work cell.

The nearest solution to an optimal design methodology was presented by Kamoun et. al, with an approach concerning the display of equipment over a given area [13]. Recently, Pelleginelli et. al, proposed an extended formalisation of design and motion planning problems for spot-welding multi-robot cells [16]. However, both solutions only considered previously selected equipment or reduced features and did not include the optimal selection of such equipment.

Despite the tremendous utility that optimisation methodologies have shown when applicable to other principals, the robotic industrial world still lacks a flexible and autonomous solution regarding the complete optimisation of work cell and its properties.

3 System Architecture and Heuristic-Based Approach

The optimisation approached presented is directly linked to an user interface software. This software follows the paradigm of MVVM (Model-View-View Model), a three layer software architecture. Each layer has its own importance. The Model is the centralized database of all relevant content. Here we can detail system components and their relations. The View Model is responsible for the communication management between the raw data on the model and the user interface that is referenced as View.

Thus, in order to use the proposed optimisation approach, each user has to follow a script of inserting work cell components, defining relations between them and then generating a robotic Kinematic Chain that will be storage as the system Model.

The View Model of the proposed approach is related to the optimisation methodology. The raw data inserted in the Model will be de-serialized and the kinematic chain's components will be classified as fix parts, conveyor, external axis, robots or tools. Then an world map is created accordingly to the found components and converted to a visual interface, View.

Once loaded the system, the optimise algorithm is fully defined. This algorithm can then be described as a set of steps (see algorithm 1).

Algorithm 1: Optimization Proceeding

Inputs: Model Data, Optimisation Algorithm;
Outputs: Optimized Pose;

```
Map = GenerateMap(Model Data);
Job = LoadJob();
Points = IdentifyPoints(Job);
OptimisedPoses = CreateStructure(Optimisation Algorithm);

RunOptimisationAlgorithm(Points,Optimisation Algorithm);
```

The terminal part of the optimisation proceeding differs accordingly to the chosen algorithm. An insight on what algorithms have been selected and validated will follow in section 4.

However, all of the selected algorithms follow an ideology of heuristic-based solutions and share a common goal, minimization of effort and maximization of present and subsequent poses. In that regard, a cost function was developed based on six features:

1. *External Axis Motion:* While performing a task, it is pretended to minimize the external axis moves as they may insert instability within the robotic system.
2. *Singularities:* Robots' behaviour becomes unstable during singularities, thus, they should be avoided.
3. *Configuration Change:* Robots should whenever possible keep an original configuration to avoid sudden uncontrolled movements.
4. *Joints' Effort:* Minimization of system effort smooths moves and protects components.
5. *Reachability:* The distance between robot's base and goal position should be minimized to increase the reaching probability for future poses.
6. *Joints' Limits:* Similar to the previous criteria, in order to increase the reaching probability of future poses, an ideal pose should maximize the interval between joint position and limits.

Thus, the cost function that untimely dictates the viability of a random pose results in a weighed sum and can be described using equation 1.

$$\begin{aligned} COST = \sum (& w_1 * ExternalMove + w_2 * Singularities \\ & + w_3 * ConfigurationChange + w_4 * JointsAmplitude \\ & + w_5 * Reachability + w_6 * JointsLimits) \end{aligned} \tag{1}$$

Although not all variables stated in the equation are not continuous (ExternalMove and ConfigurationChange), the cost function value is well defined in all its domain. That is accomplish by processing each feature separately as described below.

External Axis Motion is consider a boolean state, on whether the generated solution requires external axis moves. In that case, the value *ExternalMove* will be set to 1, otherwise it will be 0. The associated weight w_1 is fairly high due to the fact that this is one of the feature that we desire to avoid the most.

Singularities are determined and analysed using the robot Jacobian where its determinant is a indicator of singularities. When close to 0, the robot is approach a singularity state. Thus, the inverse of that value (will be higher when closer to that singularity state) is used as the *Singularities* parameter.

Each robot manufacture has a configuration definition for a given joint state. Generically this is linked with the wrist-shoulder-elbow configurations. Ideally, when finding a new solution for a specified position, robots should avoid changing configuration as it prevents uncontrolled movements. Once again, *ConfigurationChange* is a binary value, 0 when configuration change is not require, 1 otherwise.

Regarding joints' effort, the weight associated to each joint is not linear since that some joints have higher implications than other. Considering as an example, an anthropomorphic robot, the initial three joints' amplitude is more relevant than the wrist joints. Therefore the parameter *JointsAmplitude* is obtained from a weighed sum with decreasing weight for each joint starting from base to end.

The *Reachability* value is calculated based on the robot full length and is defined as the quotient between base to goal position distance and the full length value.

Finally, the joints' limits follow the same pattern as the previous parameter, as it results from a quotient between estimated joint value and its limits. However, another layer is inserted here as for each joint the weight is different for similar reasons presented for the joints' effort parameter.

One final comment is related to the usage of the cost function when there is no available solution. In that cases the testing hypothesis is discard without even being weighed.

By combining all values it is possible to classify any given pose of the system and, thus, finding the optimal solution following one of the methodologies presented next.

4 Optimisation Methodologies

To face the proposed challenge there were selected four mains optimisation techniques: Linear Scanning of available poses, Genetic Algorithms, Simulated Annealing and Potential Fields.

Each of the proposed methods returns a heap storing the best outcomes of the cost function. Considering that each solution results in an array of joint values throughout the kinematic chain, a dynamic structure is built upon the

map generation. Every heap element will follow the parameters define within that same structure.

The reason behind using a multi-solution heap is related to the continuous path of the robotic system following the array of solutions. Even if a random position is validated and optimised, along the path between poses, there might be an extra constraint such as obstacles or speed effort. Thus, in those cases, the initial best solution has to be discarded and new one will be searched within the heap.

Each method also can be divided in two main phases: creation of hypothesis and validation. Since the idea is to find the optimal robotic system pose for a pre-defined position, the hypothesis initial focus the external axis values and then using a path planner validates and determines the robot positioning for those external axis values.

The implemented methods will be synthesise in the following subsections.

4.1 Linear Scanning

The standard and easier way to do a search for an optimal solution is linearly go through all hypothesis while saving the best one. Since the final return is expected to be a heap, the saving results need to be extended to its size.

The key step of this method is selecting the discretization step that balances memory usage and time consumption. As expected this method raises problems for high redundancy systems as computational capacities are limited and considerations on memory usage need to be consider. Thus, when creating the hypothesis to test a linear discretization method for each element of interest was implemented, bounding the number of hypothesis.

The ideal algorithm can be described as following (algorithm 2).

Algorithm 2: Linear Scanning Algorithm

```
Inputs: PointsToOptimise, Model Data;
Outputs: Optimised Poses;

Hypothesis = Create_Testing_Hyphotesis();
Optimised Poses = Create_Poses_Structure();

foreach PointsToOptimise do
    foreach Hypothesis do
        EvaluteHypothesisCost(Hypothesis.Current, PointToOptimise.Current);
        Update_Optimised_Poses();
    end
end

return Optimised Poses;
```

4.2 Genetic Algorithms

This method can be defined as a search and optimisation tool able to solve multi-constraint problems [6]. Genetic algorithms recur to genes (variable of interest) to store a sequence or solution of interest. Most of common applications using this method start with two set of solutions and iteratively swap (exchange of genes between solutions) or mutate (random or methodical change of a given gene), creating a population of solutions.

Within our proposed methodology we start with a higher number of randomly generated genes. Each gene is defined as a vector resulting of the external axis values. Then, each gene undergoes a reachability validation of the defined position. Iteratively new genes are generated throughout a fixed number of iterations and the optimised heap is built. The generation of each gene is based on the swap and mutation operations, that are randomly selected. In case of swap procedure, the second gene is also randomly chosen from the multi-gene population. The algorithm can be describe as following (algorithm 3).

Algorithm 3: Genetic Algorithm

```
Inputs: PointsToOptimise, Model Data;
Outputs: Optimised Poses;

Population = Generate_Genes();
Optimised Poses = Create_Poses_Structure();

foreach PointsToOptimize do
    for NumberOfIterations do
        foreach Gene in Population do
            EvaluteHypothesisCost(Gene, PointToOptimize.Current);
            Update_Optimised_Poses();
        end
        Population = Generate_NewGenes(Population, mutationRate,
         swapRate);
    end
end

return Optimised Poses;
```

4.3 Simulated Annealing

Another optimisation technique is the simulated annealing method, which is a probabilistic to find the global optima of a given function [14]. This method starts from a random solution and iteratively searches its neighbourhood to define new possible solutions. Then, probabilistically decides to which solution it should iterate until untimely finds the global optimum.

However, when dealing with high redundancy system this method entails a high time and computation effort. Thus, a minor adjustment to the method was implemented in order to reduce the execution time of the method. A threshold was defined in order that the method runs iteratively until reaching a fixed number of solutions that verify such constraint, stopping without fully completing the algorithm, giving a secure and acceptable list of solutions while minimizing time consumption.

Another add-in was related to the initial point. Since that this is neighbour-based, if the initial point and its neighbours do not produce a valid solution, the method would stop and a erroneous value would be found. As such, the initial point is randomly fixed within half robot's length to the goal point. The algorithm is shown next.

Algorithm 4: Simulated Annealing

```
Inputs: PointsToOptimise, Model Data;
Outputs: Optimised Poses;

Solution = Generate_Initial_AcceptablePosition();
Optimised Poses = Create_Poses_Structure();

foreach PointsToOptimise do
    while SolutionNumber < IntendedSolutionNumber do
        EvaluateSolution(Solution);
        Update_Optimised_Poses();
        Neighbours = Get_Solution_Neighbours();
        Solution = Select_Next_Solution(Neighbours);
    end
end

return Optimised Poses;
```

4.4 Potential Gradient

Similar to Simulated Annealing, Potential Gradient is an algorithm based on surrounding solutions of the current iteration. However, this algorithm stops at local optimums.

The iteration direction is defined by the sum of directional derivatives framed with the optimisation function. Once determined what is the best directional vector a new solution is generated until reaching a local optimum.

Despite being computational light, this algorithm does not guarantee optimal solutions for any given problem.

5 Validation

Attempting to ensuring a multi disciplinary validation process, two work cells with different properties were modelled and inserted in a custom simulator. Those work cells are displayed below in figure 1.

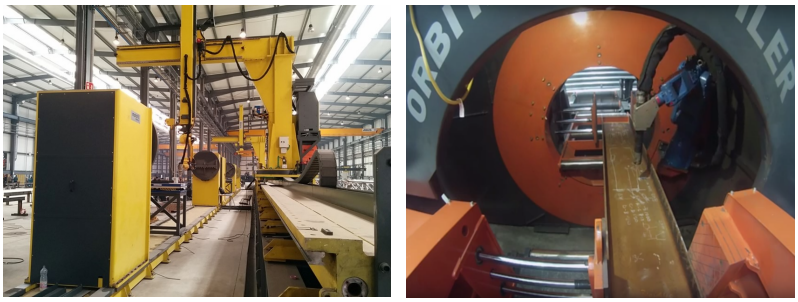

Fig. 1. At the left - a work cell with a cartesian external axis, a fanuc ic30 and a cutting torch; at the left - a work cell with two kinematic chain, one composed by a rotative external axis (Ring) and a Motoman MH5 robot and the second by a external positioner.

A third cell was used, although it is not presented above. That is due to the fact of this cell is currently being physically implemented according to the national project CoopWeld. Thus, the third cell was only test and validated in a simulated environment.

The work cells here presented are focused in these redundancy systems composed by external axis, robot and operation tool. However the proposed approach is also applicable to simpler robotic cells.

In order to validate the methodology a set of Cutting and Welding Jobs were generated using a CAM (Computer Aided Manufacturing) software for the production of beams, MetroID and CLARiSSA, proprietary of SARKKIS robotics. This software generates a set of vectors containing poses that describe the given operation (see figure 2)

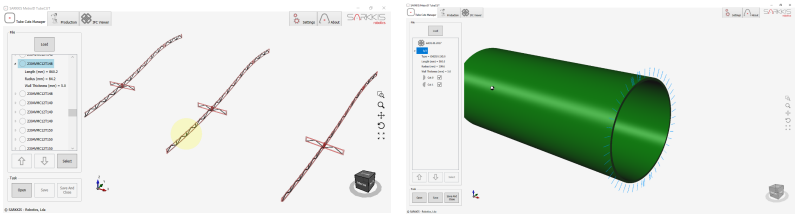

Fig. 2. MetroID user interface.

Using these software it was then created several test beams. Examples are presented next in figure 3. The idea behind the creation of those beams was to define different jobs that required external axis/positioners movement.

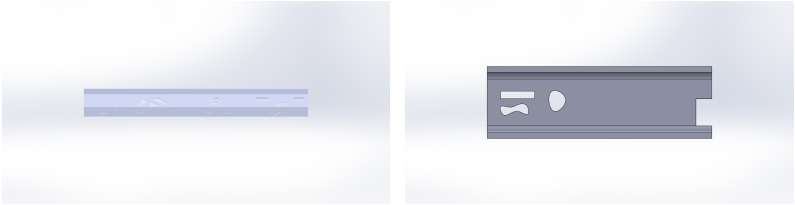

Fig. 3. Beams Examples.

In order to evaluate each of the implemented algorithms there were consider three parameters: reachability percentage, time consumption and cost of reached solution. The results are summarized in table 1. These are according to a validation test of 15 beams, each with 5 to 130 operations, which resulted in a total of 563 points to be optimised. The results provide in the table are means per operation of the correct achieved solutions.

Table 1. Results summary of the validation test

Optimisation Methodology	Solution Reachability (%)	Time consumption (s)	Cost Value
Linear Scanning	100%	18.031	0.103
Genetic Algorithms	100%	1.022	0.098
Simulated Annealing	96.4%	0.740	0.147
Potential Fields	81.4%	0.412	0.134

These results were achieved once established the proper parameters for each algorithm. Those were determined by considering the best testing performance for random positions for each algorithm when concerning memory management, time consumption and solution reachability.

Concerning Linear Scanning was implemented a discretization in 50 equally spaced hypothesis of each external axis/positioners based on their interval range. The Genetic Algorithm methodology was implemented using a cross rate of 50% and a mutation rate of 10%, for a random generated population of 1000, through-

out 25 iterations. This was the set of parameters that produced the best results in a exhaustive study done with different parametrizations. We also limited Simulated Annealing reaching goal to a maximum cost of 0.15 in order to reach a higher number of solutions. Moreover, Simulated Annealing was implemented using a multi dimensional neighbour radius of 8 increments. Each increment is considered to be the interval value of each external part when discretized into 1000 equally spaced hypothesis.

6 Discussion and Future Perspectives

In this article we presented a solution with enough flexibility to be applied to all robotic installations. This solution allows one to avoid the common pitfalls associated with robotic poses configuration.

Once defined the cost function containing key elements for poses description, a set of algorithms can be applied to identify the correct configuration for a pre-defined pose.

Here, the optimisation algorithms prove to be the best methodology. Throughout the project, four were implemented. Tests shows that each can be valuable within the scientific community.

Directional algorithms, such as Potential Gradient, proved to be faster to achieve a solution. However, they fail to reach a reasonable solution to all cases. Energy gradient based, such as Simulated Annealing, reach all solutions, however the time consumption to reach the optimal one sometimes is too high. Thus, fixing a limit may help reducing time efforts but compromises efficiency.

Linear Scanning is set to find the optimal solution within a discretization step. However, the time consumption is too high and the mentioned discretization in order to avoid memory issues might have to be enlarge reducing efficiency to find the optimal position. A recursive Linear Scanning algorithm could be implemented however it would forfeit at each iteration a set of solution (not ensuring that the optimal one is not discarded) and still remains with the time consumption problem.

Thus, Genetic Algorithms, seem to be the best methodology as they reach a viable solution for all cases with a reduce time and computation effort.

Future work concerning optimisation in robotic work cells is related to components positioning and choosing, thus, providing a complete software solution to optimal design a high redundancy robotic work cell.

In conclusion, the work presented here formalizes a flexible approach for poses optimisation methodology, ensuring optimal configuration for robotic elements to perform a given task in a faster and efficient way.

Acknowledgments

A special word to SARKKIS robotics and INESC-TEC (in particular the ERDF – European Regional Development Fund through the Operational Programme for Competitiveness and Internationalisation - COMPETE 2020 Programme,

and the FCT – Fundação para a Ciência e a Tecnologia (Portuguese Foundation for Science and Technology) within project «POCI-01-0145-FEDER-006961») for their commitment in research and development of revolutionary state-of-the-art algorithms and for their contribution regarding software tools and engineering hours availability.

References

1. Alatartsev, S., Stellmacher, S., Ortmeier, F.: Robotic task sequencing problem: A survey 80(2), 279–298 (2015)
2. Andrisano, A.O., Leali, F., Pellicciari, M., Pini, F., Vergnano, A., Pini, F.: Integrated design of robotic workcells for high quality machining (2011)
3. Bennewitz, M., Burgard, W., Thrun, S.: Optimizing schedules for prioritized path planning of multi-robot systems 1, 271–276 (2001)
4. Cheng, F.S.: Methodology for developing robotic workcell simulation models. vol. 2, pp. 1265–1271 (2000)
5. Chu, B., Jung, K., Chu, Y., Hong, D., Lim, M.T., Park, S., Lee, Y., Lee, S.U., Min Chul, K., Kang Ho, K.: Robotic automation system for steel beam assembly in building construction. pp. 38–43 (2009)
6. Deb, K.: Introduction to genetic algorithms 24(4-5), 293–315 (1999)
7. Fathi, M., Álvarez, M., Rodríguez, V.: A new heuristic-based bi-objective simulated annealing method for u-shaped assembly line balancing 10(2), 145–169 (2016)
8. Geller, E., Matthews, C.: Impact of robotic operative efficiency on profitability 209(1), 20e1–20e5 (2013)
9. Graetz, G., Michaels, G.: Robots at Work (2015)
10. Gueta, L., Chiba, R., Arai, T., Ueyama, T., Ota, J.: Compact design of work cell with robot arm and positioning table under a task completion time constraint. pp. 807–813 (2009)
11. Hauer, S., M.V.H.C.S.K.: Design and simulation of modular robot work cells. pp. 1801–1802 (2009)
12. Kamezaki, M., Hashimoto, S., Iwata, H., Sugano, S.: Development of a dual robotic arm system to evaluate intelligent system for advanced construction machinery. pp. 1299–1304 (2010)
13. Kamoun, H., Hall, N., Sriskandarajah, C.: Scheduling in robotic cells: Heuristics and cell design 47(6), 821–835 (1999)
14. Kirkpatrick, S., Gelatt Jr., C., Vecchi, M.: Optimization by simulated annealing 220(4598), 671–680 (1983)
15. Mansouri, S.: A multi-objective genetic algorithm for mixed-model sequencing on jit assembly lines 167(3), 696–716 (2005)
16. Pellegrinelli, S., Pedrocchi, N., Tosatti, L.M., Fischer, A., Tolio, T.: Multi-robot spot-welding cells for car-body assembly: Design and motion planning. Robotics and Computer-Integrated Manufacturing 44, 97 – 116 (2017)
17. Yang, J., Yang, J.: Intelligence optimization algorithms: A survey 3(4), 144–152 (2011)