Robotica (2008) volume 26, pp. 63–73. © 2007 Cambridge University Press
doi:10.1017/S0263574707003591 Printed in the United Kingdom

Application of joint error maximum mutual compensation for hexapod robots

Yauheni Veryha†* and Henrik Gordon Petersen‡

†*Department of Industrial Software and Applications, ABB Corporate Research, 68526, Ladenburg, Germany.*
‡*The Maersk Mc-Kinney Moller Institute for Production Technology, University of Southern Denmark, 5230 Odense M.*

(Received in Final Form: May 16, 2007. First published online: June 13, 2007)

SUMMARY
A good practice to ensure high-positioning accuracy in industrial robots is to use joint error maximum mutual compensation (JEMMC). This paper presents an application of JEMMC for positioning of hexapod robots to improve end-effector positioning accuracy. We developed an algorithm and simulation framework in MatLab to find optimal hexapod configurations with JEMMC. Based on a real hexapod model, simulation results of the proposed approach are presented. Optimal hexapod configurations were found using the local minimum of the infinity norm of hexapod Jacobian inverse. JEMMC usage in hexapod robots can improve hexapod end-effector positioning accuracy by two times and more.

KEYWORDS: Hexapod; Robot calibration; Positioning accuracy; Joint error maximum mutual compensation; Worst-case error.

1. Introduction

Hexapod robot belongs to the family of parallel robots, which are closed-loop mechanisms with the mobile platform connected to the base by at least two serial kinematic chains (legs). Hexapod robot is a highly rigid parallel mechanism with six degrees of freedom.[2] Parallel mechanisms, which have high force loading capacity and fine motion characteristics, were originally designed for applications as diverse as tire testing and flight simulators. Recently, researchers and developers are trying to utilise hexapods in surgical operations[3] and high-precision assembly systems, for example, in micro-handling.[11]

High accuracy of hexapod robots relies on a robust and accurate calibration of the physical robot configuration. To compensate for inaccurate robot end-effector positioning, offline generated poses[10,13] need to be corrected using a calibrated kinematic model. This leads the robot to the desired poses. Robot calibration is achieved by measuring several robot configurations and identifying the corresponding kinematic parameters. It is used to improve the absolute positioning accuracy of the end-effector by using external measuring equipment (imaging laser tracking systems, high-precision cameras, etc.).

* Corresponding author. E-mail: yauheni.veryha@de.abb.com

Conventional robot calibration techniques are heavily reliant upon numerical optimisation methods for model parameter estimation. The calibration procedure is based on the minimisation of an error function and is normally divided into two steps:[14]

- Self-calibration;
- End-effector calibration.

The purpose of self-calibration is to calibrate the closed-loop mechanism by using built-in sensors in the passive joints. After self-calibration, only the kinematic errors in the fixed transformations from the robot world frame to robot base frame and from the mobile platform frame to the end-effector frame need to be identified.[16] Representative works in self-calibration are presented by Zhuang *et al.*[23] and Yiu *et al.*[21] In[23] a self-calibration method is proposed for the conventional six-legged Stewart platform through the installation of redundant sensors in several passive joints and constructing a measurement residues with measured values and the computed values of these readable passive joint angles. When these passive joint angles are recorded at a sufficient number of measurement configurations, the actual kinematic parameters can be estimated by minimising the measurement residues. Since this model is based on the linearization of the kinematic constraint equations, it converges rapidly.[8] Iurascu and Park[12] developed a unified geometric framework for the calibration of kinematic chains containing closed loops. Both joint encoder readings and end-effector pose measurements can be uniformly included into this framework. As a result, the kinematic calibration is cast as a nonlinear constrained optimisation problem. Khalil and Besnard[14] proposed an approach for self-calibration of parallel robots without extra sensors by measuring end-effector location of the manipulator in several positions and then identifying model parameters based on the measurements.

Robot calibration based on the implicit model is a standard method to calibrate parallel robots.[15,22] For mathematical reasons, the number of equations given by the measurements in robot calibration has to be at least as large as the number of unknown parameters. Since the measurement data are usually given by sensors, it is necessary to take into account the noise associated with measuring devices. Some of the latest methods proposed by researchers attempted to handle the problem of partial information in the calibration of parallel robots by introducing semi-parametric parallel robot

https://doi.org/10.1017/S0263574707003591 Published online by Cambridge University Press

models,[5,7] which lead to low computational effort important for real applications. Daney *et al.*[9] proposed a method that gives a certified approximation of kinematic calibration. For a set of measurements given with attached uncertainties, a list of intervals for the kinematics parameters is returned, such that any solution corresponding to an instance of configuration satisfying the measurements has to belong to those intervals. This method is a new version based on interval arithmetic, using interval analysis of the so-called implicit or inverse calibration method, the most studied method for the identification of the kinematic parameters of a parallel robot.[8,23]

An error function based on the inverse kinematic model is often considered as the most numerically efficient method,[4] because an inverse kinematic model can be written in closed-form for most parallel structures, contrary to the direct kinematic model.[2] The main limitation is the necessary end-effector full-pose measurement (position and orientation of the end-effector), which have to be obtained using high-precision measuring equipment, which is usually quite expensive. The high-precision camera is one of the possible measuring devices. Not only end-effector observations, but also the observation of legs with a camera enable one to get attained poses, and then, achieve the mechanism kinematic calibration.[1,17] For many parallel mechanisms, the simultaneous observation of the mechanism legs and end-effector can be achieved.[17] As information redundancy is the basis for the error function design, the calibration efficiency should be logically improved by a simultaneous observation of all the mechanism elements.[1]

In addition to scientific aspects of robot calibration we would like to point out some of the most important requirements to robot calibration coming from practice nowadays, which include:

- Ease of use of the calibration method, which should not require special skills or too expensive equipment;
- The measurement system must be brought to the robot and not vice-versa;
- The measurement system should work under factory floor conditions and should not pose special requirements to the environment.

As already mentioned, typically, robot end-effector positioning accuracy can be improved either by precise manufacturing and assembly or by calibration of each individual robot using a kinematic model which takes errors in robot structure into account. The latter has the advantage of leading to low-cost solutions but requires sophisticated modeling of the robot structure, as described above. Another approach, which is not widely used in practice, is the effect of joint error maximum mutual compensation in robot structures.[19,20] The main objective of our research work is to apply joint error maximum mutual compensation (JEMMC) for hexapod robots and improve hexapod end-effector positioning accuracy. JEMMC can be successfully applied for hexapod and similar robots, as well as for all types of robotic manipulators with at least two rotary joints in one plane.[18] JEMMC is very easy to use in practice, because it does not require any additional equipment and satisfies most of the above-mentioned requirements to robot calibration coming from practice. JEMMC can be used after the calibration of the given robotic manipulator is finished. Thus, JEMMC is not an alternative to the calibration; it is an add-on to calibration to further improve robot end-effector positioning accuracy even if joint errors are unknown or not precisely known. JEMMC can be used for both robotic manipulators with closed-loop control systems and for robotic manipulators with open-loop control systems.

The paper is organized as follows. In Section 2, we describe the core of JEMMC. In Section 3, we show how JEMMC can be applied to hexapod robots. In Section 4, we present the Microbotic platform as a test platform for our further simulations. In Section 5, we describe our simulation algorithms and tools. In Section 6, we present and discuss our simulation results. In Section 7, we present a short summary with conclusions.

2. Joint Error Maximum Mutual Compensation (JEMMC)

It is an important issue in robotics in general to achieve high-positioning accuracy of the robot end effector. Robot end-effector positioning accuracy is defined by the industrial standard ISO 9283. High accuracy is most often achieved solely by designing the robot to be stiff and having little or no backlash. This means that joint errors are made small by design. However, to achieve say just a factor of two improvements in joint errors may increase the cost of the robot quite dramatically. Therefore, it is somewhat surprising how little efforts there have been in searching for improvements by other methods apart from calibration. The JEMMC method is applied after the calibration to compensate for the non-repetitive part of the joint errors. To be more specific, consider a fixed robot configuration, where joint encoder readings are given by some values, and where the tool position and orientation have been measured by some external system. Ideally, this configuration can then be calibrated. However, when the robot is moved around and then back to the same encoder readings, the position and orientation of the tool will be different due to the non-repetitive part of the joint error. JEMMC searches for optimal robot configurations where these unknown non-repetitive joint errors lead to as small an error as possible on the position and orientation of the end effector. This optimal configuration (or a neighbourhood around this configuration) can then be used for the required high-precision tasks. It is to be noticed that the JEMMC method can also be used without a prior calibration, in which case the joint errors must be expected to be bigger, but where one may then also have a better chance of making some vague estimates of the sign and size of these errors. It should also be noticed that there may be constraints on the search so that it is only admissible to use a limited part of the working area of the robot.

Generally, there are two possibilities to search for the optimal robot configurations with JEMMC:

- Global search in the whole robot workspace
- Local search in the given small area of the robot workspace due to the constraints mentioned above.

https://doi.org/10.1017/S0263574707003591 Published online by Cambridge University Press

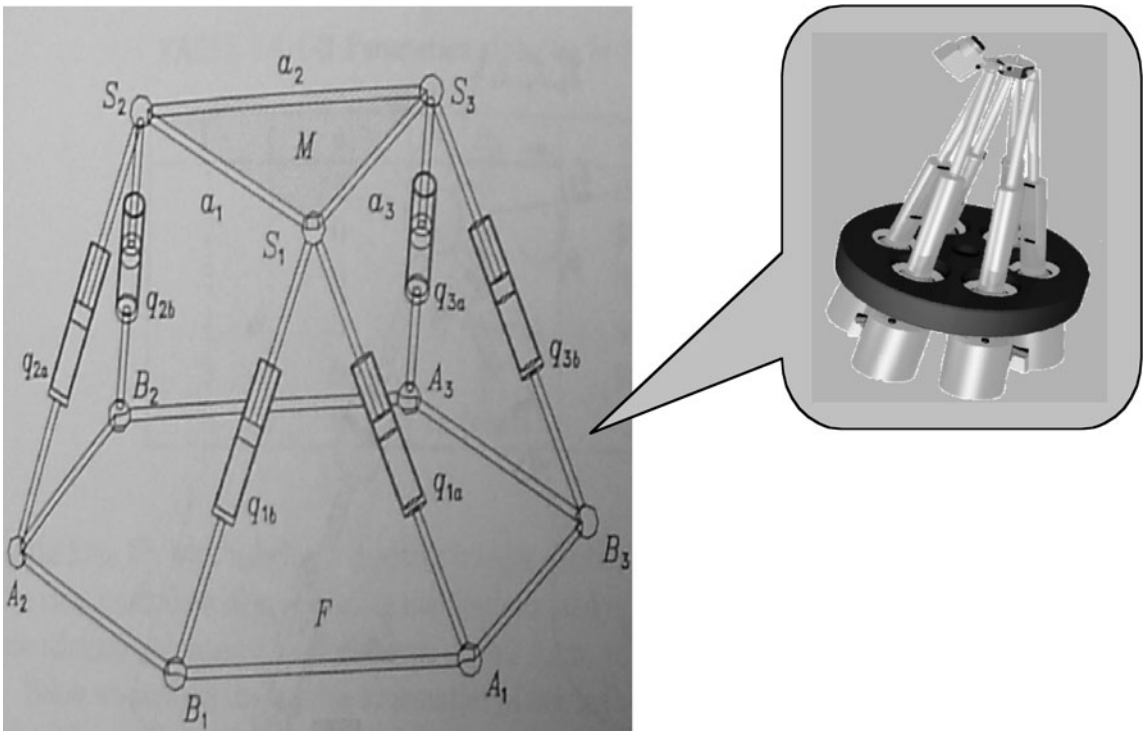


Fig. 1. Hexapod robotic manipulator and its real 3-D model.

In both cases, it is recommended that the search is done in advance, before accommodating other technological equipment and fixing assembly parts in their positions. This early investigation of the robot end-effector positioning accuracy and robot joint errors provides more flexibility in the later planning of robot operations with the optimal use of end-effector positioning accuracy.

Let us define a term *Joint Error Maximum Mutual Compensation* (JEMMC) more precisely and use it when one attempts to find the configuration of robotic manipulator in which the best end-effector positioning accuracy exists under the given joint errors $\Delta\mathbf{q}$.[20] Algorithm 1 explains JEMMC in more details.

Algorithm 1

1. Determine a part of robot workspace in which the given technological operation is to be carried out.
2. Determine the working pose $\boldsymbol{P}_{\text{initial}}$ in the given part of the robot workspace (usually, the choice of this pose is limited by the location of technological equipment).
3. Determine the allowed change a_{dist} of the location of the chosen working pose $\boldsymbol{P}_{\text{initial}}$.
4. Determine a new pose $\boldsymbol{P}_{\text{final}}$ using one of JEMMC approaches (e.g., local search, worst-case error, global search, etc.),[19,20] which is located within a_{dist} from $\boldsymbol{P}_{\text{initial}}$ in the robot workspace.
5. Plan a trajectory into pose $\boldsymbol{P}_{\text{final}}$ so that a new configuration of the robotic manipulator with the better mutual compensation of joint errors and, thus, improved end-effector positioning accuracy, is used for a given technological operation.

The key aspect in the implementation of JEMMC is the availability, unavailability or partial availability of joint-error values for a given robotic manipulator poses. As mentioned, the most interesting case for practical implementation of JEMMC is when joint-error values are not known. During our implementation of JEMMC to hexapod robots, we have therefore considered only this case.

3. Application of JEMMC to Hexapod Robots

In this section, we present an approach for implementation of JEMMC for hexapod positioning tasks. The hexapod robot belongs to the family of parallel robotic manipulators and has a relatively complex kinematics (see Fig. 1). We will present a numerical solution to find the optimal hexapod configurations with JEMMC.

We propose to find optimal hexapod configurations with JEMMC using the worst-case error approach (i.e., in which unknown big joint errors are mutually compensated in an optimal way). The six joint coordinates $\boldsymbol{q}$ of the hexapod actuator legs can be found as

$$\boldsymbol{q} = f(\boldsymbol{x}) \tag{1}$$

where $\boldsymbol{x}$ is the vector containing translation and rotation parameters of the hexapod end-effector. Thus joint errors $\Delta\boldsymbol{q}$ (Δq_1, Δq_2, Δq_3, Δq_4, Δq_5, Δq_6) in the lengths of the six actuator legs of the hexapod robot can be presented as

$$\Delta\boldsymbol{q} = \boldsymbol{J}\Delta\boldsymbol{x} \tag{2}$$

where $\boldsymbol{J}$ is the Jacobian of the hexapod robotic manipulator and $\Delta\boldsymbol{x}$ (Δx, Δy, Δz, $\Delta\varphi$, $\Delta\theta$, $\Delta\psi$) are end-effector errors. As we can see, this expression of joint errors $\Delta\boldsymbol{q}$ using the end-effector errors $\Delta\boldsymbol{x}$ looks different from that for serial robotic manipulators, where Jacobian is located on the left

https://doi.org/10.1017/S0263574707003591 Published online by Cambridge University Press

side of Eq. (2). Error vectors $\Delta \boldsymbol{x}$, $\Delta \boldsymbol{q}$ and correspondingly the Jacobian matrix $\boldsymbol{J}$ can be scaled to appropriate units to comparable values of positioning errors (Δx, Δy, Δz) and orientation errors ($\Delta\varphi$, $\Delta\theta$, $\Delta\psi$) to better estimate their effect on overall hexapod pose accuracy. Using Eq. (2), we can express $\Delta \boldsymbol{x}$ as

$$\Delta \boldsymbol{x} = \boldsymbol{J}^{-1} \Delta \boldsymbol{q}. \tag{3}$$

Unknown joint errors are joint errors, which values are not known (one can not approximate, measure or predict them, because of limited precision of sensors, imperfect error approximation models, non-repeatative errors, etc.). Now, we make the reasonable assumption that the unknown joint errors are in the interval $(-\varepsilon, \varepsilon)$ (imaginary interval within which unknown joint errors are located), which means $||\Delta \boldsymbol{q}||_\infty \leq \varepsilon$ where $||\Delta \boldsymbol{q}||_\infty$ is the infinity norm defined by $||\Delta \boldsymbol{q}||_\infty = \max_{i=1..6} |\boldsymbol{q}_\mathrm{i}|$ ($i = 1..6$, because we have six actuators in the hexapod), we can write that $\Delta L_{\max}$ (worst-case positioning accuracy of hexapod end-effector under the given joint errors) can be found as

$$\Delta L_{\max} = \max(_{||\Delta \boldsymbol{q}||_\infty \leq \varepsilon}\{||\Delta \boldsymbol{x}||\}). \tag{4}$$

To find the optimal configuration of hexapod robot with joint error maximum mutual compensation and best positioning accuracy, our task is to find the minimum of $\Delta L_{\max}$. We continue with Eq. (3) and write by definition that

$$||\Delta \boldsymbol{x}||_\infty \leq ||\boldsymbol{J}^{-1}||_\infty ||\Delta \boldsymbol{q}||_\infty \tag{5}$$

where $||\Delta \boldsymbol{x}||_\infty$ and $||\boldsymbol{J}^{-1}||_\infty$ are, accordingly, the infinity norms of Cartesian end-effector errors and inverse Jacobian. Taking into account that $||\Delta \boldsymbol{q}||_\infty \leq \varepsilon$, we then get

$$\Delta L_{\max} \approx \varepsilon ||\boldsymbol{J}^{-1}||_\infty. \tag{6}$$

Thus, to find the minimum of $\Delta L_{\max}$ we have to find the minimum of infinity norm $||\boldsymbol{J}^{-1}||_\infty$ (it follows from theory of matrices and properties of the infinity norm for matrices[2])

$$\min(||\boldsymbol{J}^{-1}||_\infty). \tag{7}$$

To show how to solve Eq. (7), we write the elements of the inverse of hexapod Jacobian $\boldsymbol{J}^{-1}$ as:

$$\boldsymbol{J}^{-1} = \begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} & a_{15} & a_{16} \\ a_{21} & a_{22} & a_{23} & a_{24} & a_{25} & a_{26} \\ a_{31} & a_{32} & a_{33} & a_{34} & a_{35} & a_{36} \\ a_{41} & a_{42} & a_{43} & a_{44} & a_{45} & a_{46} \\ a_{51} & a_{52} & a_{53} & a_{54} & a_{55} & a_{56} \\ a_{61} & a_{62} & a_{63} & a_{64} & a_{65} & a_{66} \end{bmatrix}, \tag{8}$$

where the elements a_{ij} depend on the robot configuration and are found by inverting the matrix $\boldsymbol{J}$. We do not show the details of elements a_{ij} because expressions formed after inversion of (8) would require too much space.

By definition, the infinity norm $||\boldsymbol{J}^{-1}||_\infty$ is[6]

$$||\boldsymbol{J}^{-1}||_\infty = \max_{i=1..6} \sum_{k=1}^{6} |a_{ik}| \tag{9}$$

If we define sums $\mathrm{A_i}$ for each of matrix rows as

$$\mathrm{A_i} = \sum_{k=1}^{6} |a_{ik}|, \qquad i = 1..6$$

then

$$||\boldsymbol{J}^{-1}||_\infty = \max(\mathrm{A_1}, \mathrm{A_2}, \mathrm{A_3}, \mathrm{A_4}, \mathrm{A_5}, \mathrm{A_6}). \tag{10}$$

We may now formally write $\mathrm{A_i} = f_\mathrm{i}(x, y, z, \varphi, \theta, \psi)$. At this point, we may use a standard optimisation routine to find the minimum of the infinity norm $||\boldsymbol{J}^{-1}||_\infty$. Based on[6] among quasi-Newton methods the BFGS (Broyden, Fletcher, Goldfarb, and Shanno) algorithm is among the most successful as it is one of the fastest and shows good convergence properties. We have therefore decided to use the BFGS method.

Based on Algorithm 1, we now summarise by presenting a more elaborate Algorithm 2.

Algorithm 2

1. Determine the working pose $\boldsymbol{P}_{\mathrm{initial}}$ $(x, y, z, \varphi, \theta, \psi)$ for the hexapod end-effector in the given part of the robot workspace (usually, the choice of this pose is limited by the location of technological equipment).
2. Calculate the Jacobian matrix $\boldsymbol{J}$ of hexapod for $\boldsymbol{P}_{\mathrm{initial}}$ $(x, y, z, \varphi, \theta, \psi)$.
3. Calculate sums for each of rows $\mathrm{A_i}$ (i = 1 .. 6) based on the Jacobian matrix $\boldsymbol{J}$ values.
4. Identify maximum between $\mathrm{A_i}$.
5. Use the maximum $\mathrm{A_i} = f_\mathrm{i}(x, y, z, \varphi, \theta, \psi)$ found on step 4 as the infinity norm $||\boldsymbol{J}^{-1}||_\infty$, and minimise its $f_\mathrm{i}(x, y, z, \varphi, \theta, \psi)$ using the BFGS algorithm.
   a. On each step of BFGS algorithm check if $\mathrm{A_i}$ is still an infinity norm $||\boldsymbol{J}^{-1}||_\infty$. If yes, then continue minimisation. If no, then remember current coordinates $(x, y, z, \varphi, \theta, \psi)$ as critical ones (these coordinates can be later used as intermediate "success", if the minimisation process does not converge further for any reasons) and go to step 4 to continue minimisation with another $\mathrm{A_i}$ which became the infinity norm $||\boldsymbol{J}^{-1}||_\infty$.
   b. Continue minimisation until the local minimum for infinity norm $||\boldsymbol{J}^{-1}||_\infty$ is found with the required level of accuracy. This means the coordinates of the new pose $\boldsymbol{P}_{\mathrm{final}}$ $(x', y', z', \varphi', \theta', \psi')$ will become available.

Algorithm 2 is the variation of Algorithm 1 as applied to hexapod robots. If you would refer to software development, then Algorithm 1 is the declaration of JEMMC and Algorithm 2 is the implementation of JEMMC for hexapod robots. We omitted the allowed change a_{dist} of the location of the chosen working pose $\boldsymbol{P}_{\mathrm{initial}}$ in our implementation of

https://doi.org/10.1017/S0263574707003591 Published online by Cambridge University Press

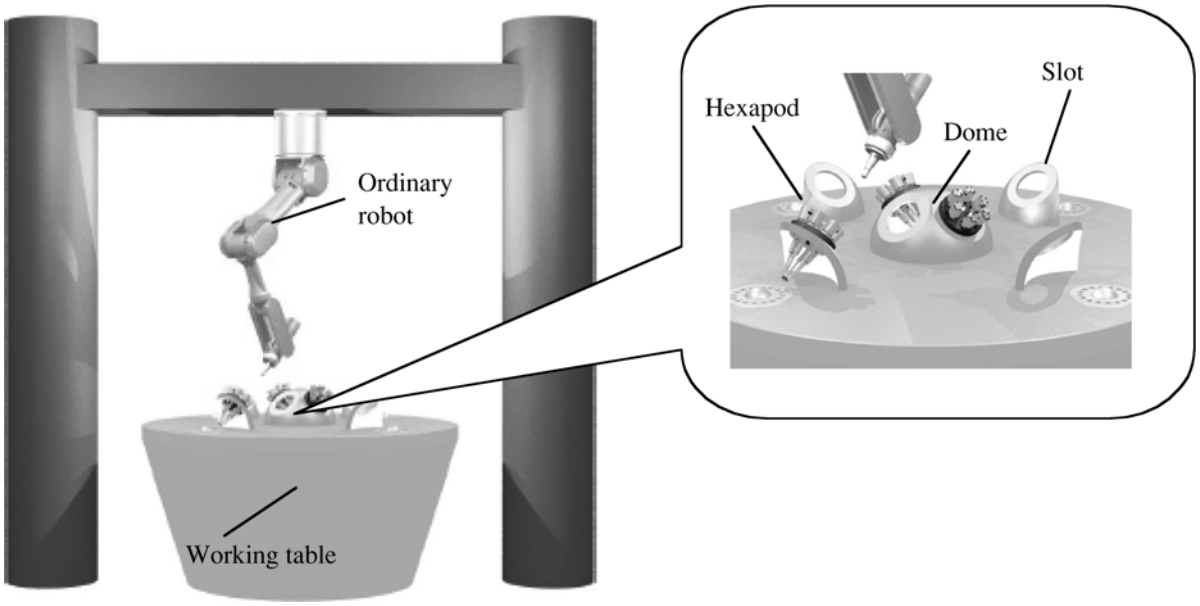


Fig. 2. Multiple hexapod robots brought together to solve complex micro-handling task in the Microbotic platform.

JEMMC in Algorithm 2 to avoid a more compex constrained optimisation in our solution.

It is important to mention that we have only considered joint errors of the actuated hexapod leg lengths. Joint errors in the passive joints of the hexapod legs can be readily found if joint errors in active joints are known.

4. Microbotic Platform as a Real Example

Although the JEMMC method presented in the previous section is actually applicable for practically all hexapod types and many other parallel mechanisms, the main example considered in this paper is an application to the Microbotic platform.[11] The Microbotic platform is a fully-automated hexapod based platform for micro-handling and processing of micro-parts, for example, in hearing-aid assembly. The Microbotic platform (see Fig. 2) comprises a number of hexapods, slots, an ordinary robot, part feeders, and a so-called dome, which is used to delimit a part of the workspace from the surrounding environment and to provide slots for hexapod robots.[11]

Slots provide mounting points for hexapods, and only when mounted in a slot, a hexapod will be able to move. The hexapods are mobile in the sense that it is possible to move them between different slots while the workcell is working. By using a miniature version of the hexapod (see Fig. 3), we get the necessary dexterity to perform arbitrary motions in 3-D space and the rigidity to obtain the accuracy that we need. An ordinary industrial robot hanging over the workcell suspended by an iron portal takes care of hexapod translocations. Inside the dome, it is possible to control environmental parameters such as light, temperature, dust and atmosphere, which quite often constitute important factors in micro-handling and processing. Complex micro-handling can, thus, be done in the shelter of the dome by bringing together multiple hexapod robots for co-operation.[11]

Obtaining the necessary accuracy is a difficult task requiring not only development of highly specialized and

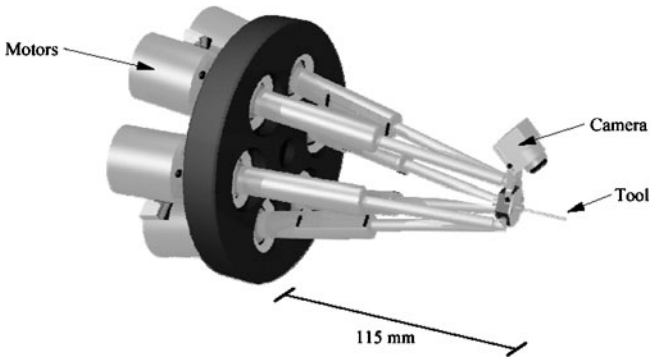


Fig. 3. Miniature hexapod from the Microbotic platform.[11]

accurate mechanics but also the advanced control. The control of the hexapod relies on the feedback provided by an unusual combination of sensor systems.[11] In short, two different types of sensors are used. An external optical sensor system that measures the pose of the tool of the hexapod relative to some global workcell reference. In addition, a little video camera is mounted near the tool of the hexapod, as shown in Fig. 3. When the hexapod approaches the expected position of a certain workpiece, the object enters the field of camera view. Using advanced computer vision algorithms, the video stream is analyzed and the pose of the tool relative to relevant features of the workpiece is calculated.[11] These two measurements work in conjunction to provide highly accurate probing of the pose of the hexapod tool in order to enable relevant control.

Our proposal for such applications like the Microbotic platform is to use JEMMC for hexapod robots. This means the technological operations will have to be executed by hexapods in such configurations (or in close neighborhoods of those configurations) in which the best joint error mutual compensation takes place, as it was discussed in previous sections. This will provide higher accuracy of the technological operation execution by the Microbotic platform.

https://doi.org/10.1017/S0263574707003591 Published online by Cambridge University Press

In this paper, we show an example of JEMMC application to only one hexapod robot. In our future works, we may present how JEMMC can be applied to more than one hexapod robots in the Microbotic platform to show the effect of JEMMC in a complex environment with multiple robots.

5. Simulation Approach

To calculate the minimum of the infinity norm $||\boldsymbol{J}^{-1}||_\infty$ and take into account limitations of the hexapod robot workspace, we have as mentioned chosen to use the BFGS algorithm. The implementation of a general constrained optimisation algorithm is in most cases quite complex. Thus, we have decided to limit our task in this work to use BFGS unconstrained optimisation algorithms from MatLab Optimisation Toolbox. Thus, we developed the MatLab module for minimisation of the infinity norm $||\boldsymbol{J}^{-1}||_\infty$ using the unconstrained BFGS algorithm. If the unconstrained optimisation algorithm gets stuck, the problem can be solved using selection of proper initial conditions. In practice, one can use our developed MatLab module as shown in the scheme in Fig. 4. The module for minimisation of the infinity norm $||\boldsymbol{J}^{-1}||_\infty$ can be accessed using the MatLab Application Program Interface (API) by the trajectory generator of robot control system.

MatLab API is a library that allows writing C and Fortran programs that interact with MatLab. It includes facilities for calling routines from MatLab (dynamic linking), calling MatLab as a computational engine and for reading and writing MAT-files. As it was mentioned in Algorithm 2, hexapod trajectory generator will provide the coordinates (translation and rotation parameters) of working pose $\boldsymbol{P}_{\text{initial}}$ of hexapod end-effector for a given technological operation to the developed MatLab module to find the improved position $\boldsymbol{P}_{\text{final}}$ based on the minimisation of the infinity norm $||\boldsymbol{J}^{-1}||_\infty$. Leading to the (locally) optimal end-effector pose $\boldsymbol{P}_{\text{final}}$, in which better end-effector positioning accuracy can be expected because of the improved joint error mutual compensation. The hexapod trajectory generator can plan then a new trajectory into the newly found $\boldsymbol{P}_{\text{final}}$, if it is allowed under the given circumstances (e.g., limitations from technological equipment, robot workspace etc.).

It should be mentioned that our solution in Matlab is not capable of providing real-time calculations for online hexapod trajectory corrections to improve end-effector positioning accuracy. It takes about 30 s to find a minimum of $||\boldsymbol{J}^{-1}||_\infty$ using IBM Laptop (2 MB RAM, 20 GB free hard disk space and ~1.6 GHz CPU). One could develop the same optimisation functionality in C or C++, but this would be very time-consuming taking into account that one had to develop a very stable BFGS method implementation in C++ for:

- Handling of very complex functions for which one cannot easily obtain gradients (e.g., in Matlab one can use a built-in method of finite differences to approximate the function gradient)
- Support for minimisation of functions, which include such complex operations as inversion of matrices, infinity matrix norms, etc.

We think that for research purposes, the existing Matlab-based implementation is good enough to test the most important characteristics of the developed optimisation framework for hexapod robots.

6. Simulation Results

Here, we present some representative set of our simulation results. We defined parameters of our hexapod robot from Microbotic platform[11] by setting radii of the hexapod base and end-effector to $r_b = 0.035\,\text{m}$ and $r_e = 0.01\,\text{m}$, accordingly, height of the platform in neutral position to $h = 0.115\,\text{m}$ and angles between the first two joints on the

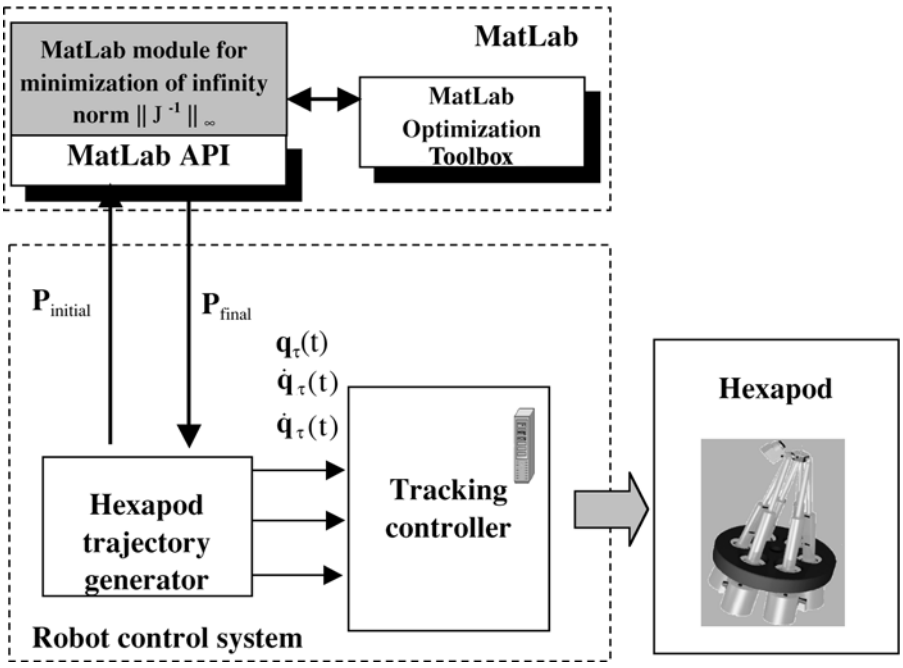


Fig. 4. Application of MatLab module for minimisation of infinity norm $||\boldsymbol{J}^{-1}||_\infty$.

https://doi.org/10.1017/S0263574707003591 Published online by Cambridge University Press

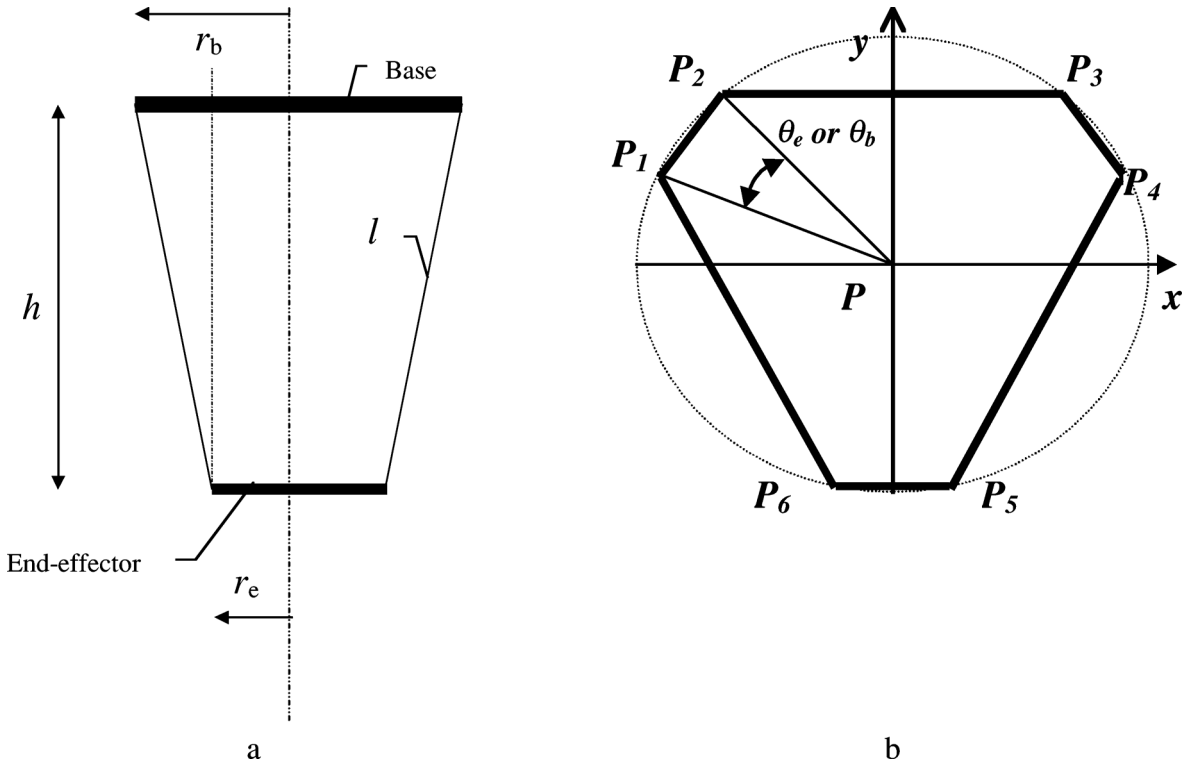


Fig. 5. Geometry of hexapod robot used in simulations. (a) height h in neutral position, leg length l and radii r_b , r_e of hexapod base and end-effector (side view); and (b) angles θ_b and θ_e between the first two joints on the base and end-effector (front view).

base and end-effector to $\theta_b = 30^\circ$ and $\theta_e = 0^\circ$, accordingly (see Fig. 5 for details). The lengths of hexapod legs in the neutral position were found to be $l = 0.12$ m. The allowed range of hexapod leg lengths is [0.109 . . . 0.139 m]. Such a hexapod robot has the hexagon as the base and triangular end-effector (see Fig. 1 and Fig. 5).

We defined an initial hexapod end-effector pose (freely and randomly selected in the hexapod robot workspace so that leg lengths are within [0.109 . . . 0.139] m) with coordinates $\boldsymbol{P}_{\text{initial}}\,(x, y, z, \varphi, \theta, \psi)$ for our unconstrained hexapod robot simulation

$$\boldsymbol{P}_{\text{initial}}(-0.07\,\text{m}, -0.005\,\text{m}, 0.1\,\text{m}, 0.174\,\text{rad}, 0\,\text{rad}, 0\,\text{rad}), \quad \text{Infinity norm: } 1394.59$$

in which the minimisation of the infinity norm of hexapod Jacobian inverse was initialized. After running Algorithm 2 for unconstrained hexapod robot (i.e. with no limitations on leg lengths), but with constant orientation (i.e. $\varphi = \text{const}$, $\theta = \text{const}$, $\psi = \text{const}$), we obtained the following values of $\boldsymbol{P}_{\text{final}}$ $(x', y', z', \varphi', \theta', \psi')$

$$\boldsymbol{P}_{\text{final}}(-0.0036\,\text{m}, 0.0008\,\text{m}, -0.0328\,\text{m}, 0.174\,\text{rad}, 0\,\text{rad}, 0\,\text{rad}), \quad \text{Infinity norm: } 469.16$$

The process of the infinity norm minimisation for hexapod Jacobian inverse is shown in the graph in Fig. 6.

In our example, the minimum of infinity norm of hexapod Jacobian inverse was successfully found. The value of the infinity norm was significantly reduced by 66% in $\boldsymbol{P}_{\text{final}}$. To verify that positioning and orientation accuracy in pose $\boldsymbol{P}_{\text{final}}$ is better than that in $\boldsymbol{P}_{\text{initial}}$, we will find positioning and orientation accuracy (ISO 9283) in $\boldsymbol{P}_{\text{initial}}$ and $\boldsymbol{P}_{\text{final}}$ and compare them. Positioning accuracy ΔL_p can be expressed as (ISO 9283)

$$\Delta L_p = \sqrt{(\underline{x} - x_c)^2 + (\underline{y} - y_c)^2 + (\underline{z} - z_c)^2} \qquad (11)$$

where $\underline{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$, $\underline{y} = \frac{1}{n}\sum_{i=1}^{n} y_i$ and $\underline{z} = \frac{1}{n}\sum_{i=1}^{n} z_i$ are average values of the Cartesian coordinates of the attained point after n iterations; x_c, y_c and z_c are coordinates of the commanded (reference) point in the robot workspace; x_i, y_i and z_i are Cartesian coordinates of the attained points in the robot workspace obtained at iteration i and n is the number of iterations. The orientation accuracy ΔL_o can be similarly presented (see ISO 9283) using end-effector orientation angles φ, θ and ψ (roll, pitch and yaw).

We used ten random combinations (see Table I) of joint errors $\Delta\boldsymbol{q}$ to verify that positioning and orientation accuracy in pose $\boldsymbol{P}_{\text{final}}$ is better than that in $\boldsymbol{P}_{\text{initial}}$. Comparison results for JEMMC application to hexapod robots with various possible joint errors $\Delta\boldsymbol{q}$ (see Table I) are presented in Table II. As one can see from results in Table II, end-effector positioning and orientation accuracy was significantly improved in $\boldsymbol{P}_{\text{final}}$ in most cases (e.g., positioning accuracy was improved by 48% and orientation accuracy was improved by 18% on average), because of the improved joint error mutual compensation in poses with smaller infinity norm of hexapod Jacobian inverse.

However, for some combinations of joint errors (see rows 5 and 6 in Table II) the deterioration of end-effector positioning and/or orientation accuracy was observed, as expected. The deterioration of end-effector accuracy takes place when joint

https://doi.org/10.1017/S0263574707003591 Published online by Cambridge University Press

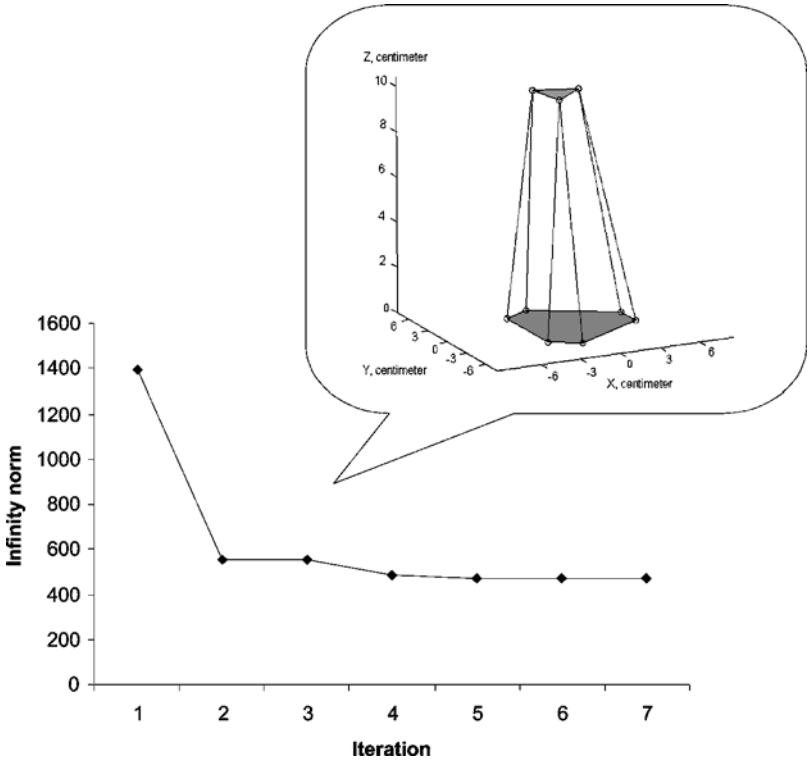


Fig. 6. Minimisation of infinity norm of hexapod Jacobian inverse using BFGS algorithm in $\boldsymbol{P}_{\text{initial}}$ (after 7 iterations and 132 function evaluations).

Table I. Ten random combinations of joint errors $\Delta\boldsymbol{q}$ (Δq_1, Δq_2, Δq_3, Δq_4, Δq_5, Δq_6) in six hexapod legs.

Combination	Δq_1 [m]	Δq_2 [m]	Δq_3 [m]	Δq_4 [m]	Δq_5 [m]	Δq_6 [m]
$\Delta\boldsymbol{q}_1$	0.0001	−0.0005	0.0005	−0.0004	0.0002	0.0003
$\Delta\boldsymbol{q}_2$	−0.0008	−0.0001	0.00004	−0.0001	−0.0007	−0.0006
$\Delta\boldsymbol{q}_3$	−0.0001	−0.0007	−0.00005	−0.00004	−0.0009	−0.0003
$\Delta\boldsymbol{q}_4$	0.0001	0.0005	0.0005	0.0009	0.0008	0.0009
$\Delta\boldsymbol{q}_5$	0.0001	0.0001	0.0001	0.0001	0.0001	0.0001
$\Delta\boldsymbol{q}_6$	−0.0002	−0.0002	−0.0001	−0.0001	−0.0001	−0.0001
$\Delta\boldsymbol{q}_7$	0.0002	−0.0002	0.0004	0.0009	−0.00002	0.000003
$\Delta\boldsymbol{q}_8$	−0.0005	−0.0005	−0.0005	0.0004	0.0004	0.0004
$\Delta\boldsymbol{q}_9$	0.0000001	−0.000005	0.000002	−0.0000004	0.000001	0.0008
$\Delta\boldsymbol{q}_{10}$	−0.00001	−0.0000005	−0.00005	−0.00002	0.0002	0.00002

Table II. Comparison of positioning and orientation accuracy in $\boldsymbol{P}_{\text{initial}}$ and $\boldsymbol{P}_{\text{final}}$ for hexapod robot.

Joint errors	Initial positioning accuracy (m) $\times 10^{-6}$	Final positioning accuracy (m) $\times 10^{-6}$	Positioning improvement/ deterioration (%)	Initial orientation accuracy (rad) $\times 10^{-3}$	Final orientation accuracy (rad) $\times 10^{-3}$	Orientation improvement/ deterioration (%)
$\Delta\boldsymbol{q}_1$	4647	1398	69.9	370	164	55.6
$\Delta\boldsymbol{q}_2$	3468	1216	64.9	199	72	63.7
$\Delta\boldsymbol{q}_3$	5639	1627	71.1	152	155	−1.8
$\Delta\boldsymbol{q}_4$	1807	1011	44.0	252	72	71.3
$\Delta\boldsymbol{q}_5$	102	126	−23.7	0.1	0.4	−155.3
$\Delta\boldsymbol{q}_6$	136	167	−23.2	7.1	8.4	−18.0
$\Delta\boldsymbol{q}_7$	3900	1263	67.6	125	129	−2.7
$\Delta\boldsymbol{q}_8$	4710	1479	68.6	238	80	66.2
$\Delta\boldsymbol{q}_9$	4149	1220	70.6	263	116	56.0
$\Delta\boldsymbol{q}_{10}$	1081	310	71.3	51	28	44.4
$\Sigma/\mathbf{10}$			**∼48**			**∼18**

https://doi.org/10.1017/S0263574707003591 Published online by Cambridge University Press

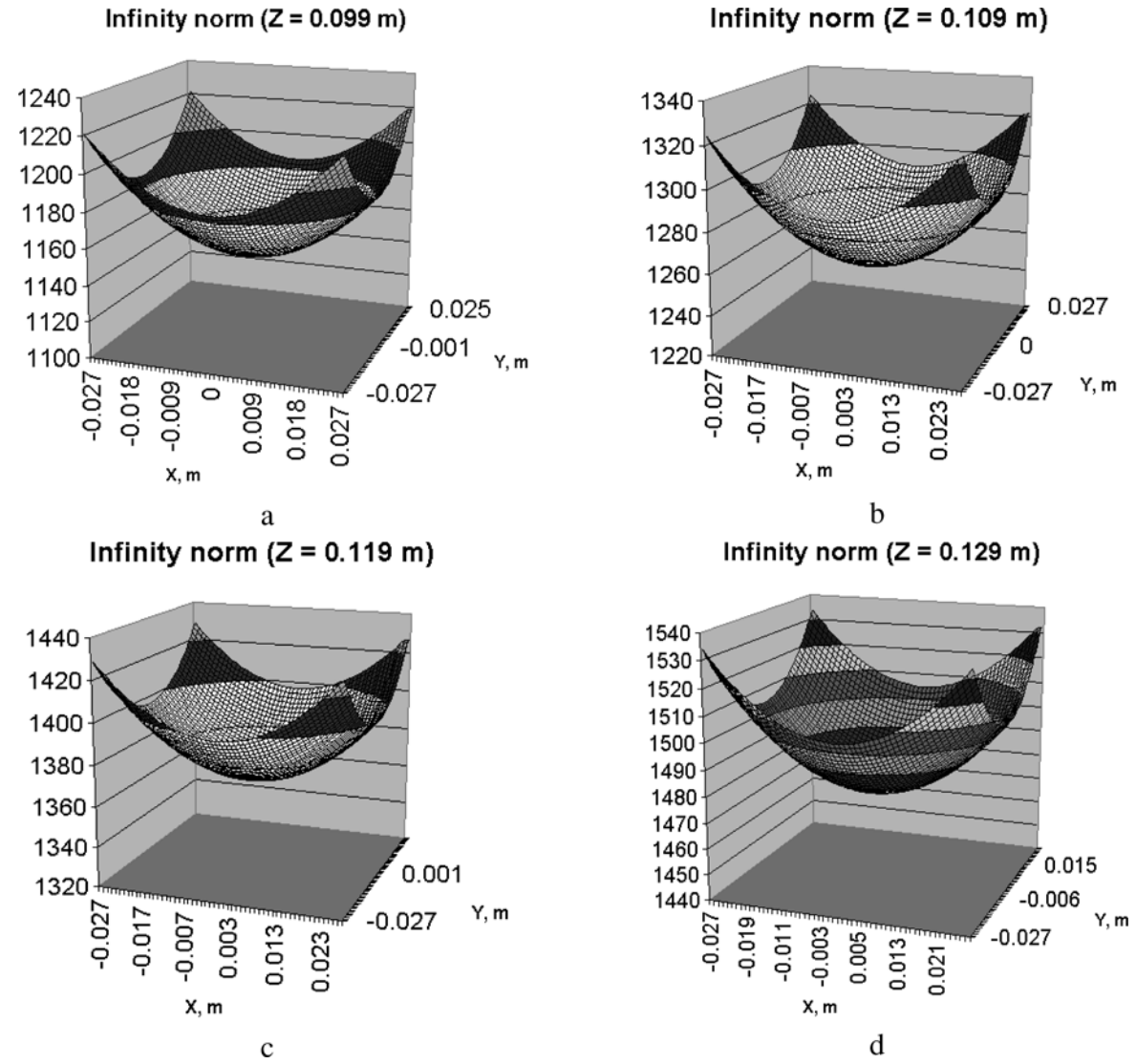


Fig. 7. Dependences of the infinity norm of hexapod Jacobian inverse on end-effector Cartesian coordinates under various Z coordinates.

errors are very small (see Tables I and II). On the opposite, the best improvement of end-effector accuracy takes place when the initial joint errors are big. This is exactly as we expected, because the presented minimisation of Jacobian infinity norm was designed for worst-case errors.

As one can see in the graph in Fig. 6, the first jump in the BFGS algorithm is quite big and one can expect that in real hexapod models the allowed range of leg length will be exceeded already on the first step of BFGS algorithm (as it was the case in our example). Thus, it is highly recommended to use one of the constrained optimisation algorithms[6] for real hexapod robots in practice.

We scanned our real hexapod workspace with the unchanged hexapod end-effector orientation (0.174, 0, 0 rad) by changing the values of end-effector Cartesian coordinates x $[-0.08 \ldots 0.08\,\text{m}]$ with $\Delta x = 0.001\,\text{m}$, y $[-0.08 \ldots 0.08\,\text{m}]$ with $\Delta y = 0.001\,\text{m}$ and z $[0.08 \ldots 0.139\,\text{m}]$ with $\Delta z = 0.01\,\text{m}$. We found a near-optimal global minimum of the infinity norm of the Jacobian inverse to be in the pose $\boldsymbol{P}'_{\text{final}}$

$$\boldsymbol{P}'_{\text{final}}(0.002\,\text{m}, 0.004\,\text{m}, 0.109\,\text{m}, 0.174\,\text{rad}, 0\,\text{rad}, 0\,\text{rad}), \quad \text{Infinity norm: } 1256.11$$

with hexapod leg lengths (0.1131, 0.1129, 0.1121, 0.1112, 0.1113, 0.1116 m). The value of the infinity norm 1256.11 is much bigger than that in our initial simulation (infinity norm = 469.16), because leg length ranges of the real hexapod were exceeded in the first simulation case.

In our latest simulation with real hexapod, the value of the infinity norm of hexapod Jacobian inverse was reduced by ~10% in pose $\boldsymbol{P}'_{\text{final}}$. Both positioning and orientation accuracies could improve in $\boldsymbol{P}_{\text{final}}$ by 21% and 16%, respectively, if joint errors would have values of $\Delta \boldsymbol{q}_1$ (see Table I) (i.e., new positioning accuracy in $\boldsymbol{P}'_{\text{final}}$ could be 3672×10^{-6} m and new orientation accuracy in $\boldsymbol{P}'_{\text{final}}$ could be 312×10^{-3} rad). Simulation results of hexapod workspace scanning to find the near-optimal infinity norm of the hexapod Jacobian inverse are shown in graphs in Fig. 7 (please note that some of the poses shown in the graphs in Fig. 7 are not within the constrained hexapod robot workspace; we needed those poses to properly build the graphs in Microsoft Excel). The minimum of the infinity norm was 1256.11 and the maximum was 1509.36 in the real workspace (i.e., leg lengths [0.109 . . . 0.139] m) of our hexapod robot from Microbotic platform. This means that if we would select our $\boldsymbol{P}_{\text{initial}}$

https://doi.org/10.1017/S0263574707003591 Published online by Cambridge University Press

in the pose with the maximum infinity norm 1509.36, and compare it with $\boldsymbol{P}'_{\text{final}}$, then we could see that the infinity norm would be reduced by 17% in $\boldsymbol{P}'_{\text{final}}$. Hence, one could expect positioning and orientation accuracy improvement by up to 30% in $\boldsymbol{P}'_{\text{final}}$.

The obtained results confirmed that joint error maximum mutual compensation was quite effective to improve end-effector positioning accuracy (e.g., one could improve positioning accuracy of unconstrained and constrained hexapod robots by 50% and 20%, respectively). The important aspect for JEMMC in conventional (constrained) hexapod robots is the range of leg lengths, for example the leg length range in hexapods from Microbotic platform may be a bit too strict.

This did not allow us using the whole power of JEMMC and significantly (by 50% and more) improve positioning and orientation end-effector accuracy in hexapod robots. If the bottom limit of leg length could be reduced from 0.109 m to 0.06 m, then the effect of JEMMC would provide the improvement of end-effector positioning accuracy for hexapod robots from the Microbotic platform by 50% and more.

7. Conclusion

In our research work, we developed the algorithm and simulation framework in MatLab to find optimal hexapod configurations with joint error maximum mutual compensation in which the positioning and orientation accuracy of hexapod end-effector is the best according to the worst-case error approach. The optimal hexapod configurations were found using the local minimum of the infinity norm of hexapod Jacobian inverse. The use of joint error maximum mutual compensation in hexapod robots can improve hexapod end-effector positioning accuracy by two times and more.

The changing values of joint errors remain the main open issue. In practice, it is still possible that in robot configurations with minimum infinity norm of Jacobian inverse joint errors may significantly increase. This may deteriorate the effect of JEMMC. It will also significantly complicate any experimental measurements to verify JEMMC in practice.

In practice, one may need to use the whole workspace of the robotic manipulator. To benefit from the JEMMC in such cases, we propose to use a slight change (e.g., up to 10–30 mm) of the end-effector working pose that allows performing local optimisation of the end-effector positioning accuracy using JEMMC. The presented JEMMC approach can find its practical implementation in robotic systems with hexapods and other similar robotic manipulators. In particular, we think that JEMMC can be widely used on the stages of robot end-effector trajectory planning.

During our work on JEMMC the following major issues were identified as subjects for future works and improvements:

- Integration of constrained optimisation algorithms for joint error maximum mutual compensation to robot off-line programming systems;
- Influence of JEMMC on robot manipulability, maximum consumption of energy, etc.; and
- Evaluation of application and usability of JEMMC not only for end-effector positioning, but also for robot path tracking.

References

1. N. Andreff, A. Marchadier and P. Martinet, "Vision-based control of a Gough-Stewart parallel mechanism using legs observation", *Proceeding the IEEE International Conference on Robotics and Automation*, Barcelona, Spain (2005), pp. 2546–2551.
2. J. Angeles, *Fundamentals of Robotic Mechanical Systems*. (Springer, New York, USA, 2003).
3. S. Bai and M. Teo, "Kinematic calibration and pose measurement of a medical parallel manipulator by optical position sensors", *J. Robot. Syst.* **20**(4), 201–209 (2003).
4. S. Besnard and W. Khalil, "Identifiable parameters for parallel robots kinematic calibration", *Proceeding of IEEE International Conference on Robotics and Automation* (Seoul, South Korea (2001) pp. 2859–2866.
5. K-S., Chai, K. Young and I. Tuersley, "A practical calibration process using partial information for a commercial Stewart platform", *Robotica*, **20**(3), 315–322, (2002).
6. E. Chong and S. Zak, *An Introduction to Optimization*. (Wiley, New York, USA, 2001).
7. D. Daney and I. Emiris, "Robust parallel robot calibration with partial information", *Proceeding of IEEE International Conference on Robotics and Automation*, Seoul, South Korea (2001) pp. 3262–3267.
8. D. Daney, "Kinematic calibration of the Gough platform", *Robotica*, **21**(6), 677–690 (2003).
9. D., Daney, Y. Papegay and A. Neumaier, "Interval methods for certification of the kinematic calibration of parallel robots", *Proceeding of IEEE International Conference on Robotics and Automation*, New Orleans, USA (2004), pp. 1913–1918.
10. D., Daney, Y. Papegay and B. Madeline, "Choosing measurement poses for robot calibration with the local convergence method and Tabu search", *Int. J. Robot. Research*, **24**(6), 501–518 (2005).
11. K. Doering, "Control and coordination of robots in a flexible multi-robot platform with a special reference to micro-handling", *Industrial Ph.D. Dissertation*, (University of Southern Denmark, Odense, Denmark, 2004).
12. C. Iurascu and F. Park, "Geometric algorithms for closed chain kinematic calibration", *Proceedind of IEEE Conference on Robotics and Automation*, Detroit, USA (1999) pp. 1752–1757.
13. Z. Ji and Z. Li, "Identification of placement parameters for modular platform manipulators", *J. Robot. Sys.* **16**(4), 227–236 (1999).
14. W. Khalil and S. Besnard, "Self calibration of Stewart-Gough parallel robot without extra sensors", *IEEE Trans. Robot. Autom.* **15**(6), 1116–1121 (1999).
15. A., Nahvi, J. Hollerbach and V. Hayward, "Calibration of a parallel robot using multiple kinematics closed loops", *Proceeding of IEEE International Conference on Robotics and Automation*, San Diego, USA (1994) pp. 407–412.
16. A. Rauf and J. Ryu, "Fully autonomous calibration of parallel manipulators by imposing position constraint", *Proceeding of IEEE International Conference on Robotics and Automation*, Seoul, South Korea (2001), pp. 2389–2394.
17. P., Renaud, N., Andreff, F. Pierrot and P. Martinet, "Combining end-effector and legs observation for kinematic calibration of parallel mechanisms", *Proceeding of IEEE International Conference on Robotics and Automation*, New Orleans, USA (2004) pp. 4116–4121.
18. Y. Veryha, *Positioning of Robotic Manipulator End-Effector Using Joint Error Maximum Mutual Compensation*. (Logos-Verlag, Berlin, Germany, 2006).

https://doi.org/10.1017/S0263574707003591 Published online by Cambridge University Press

19. Y. Veryha and J. Kurek, "Application of joint error mutual compensation for robot end-effector pose accuracy improvement", *J. Intell. Robot. Syst.* **36**(3), 315–329, (2003).
20. Y. Veryha and J. Kurek, "Robotic manipulator pose accuracy improvement using joint error maximum compensation," *Proceeding of IEEE International Conference on Methods and Models in Automation and Robotics*, Szczecin, Poland (2002) pp. 993–998.
21. Y., Yiu, J. Meng and Z. Li, "Auto-calibration for a parallel manipulator with sensor redundancy," *Proceeding of IEEE International Conference on Robotics and Automation*, Taipei, Taiwan (2003) pp. 3660–3665.
22. H. Zhuang and Z. Roth, "Method for kinematic calibration of Stewart platforms", *J. Robo. Syst.* **10**(3), 391–405 (1993).
23. H., Zhuang, J. Yan and O. Masory, "Calibration of Stewart platform and other parallel manipulators by minimizing inverse kinematic residuals", *J. Robot. Syst.* **15**(7), 396–406, (1998).

https://doi.org/10.1017/S0263574707003591 Published online by Cambridge University Press