Robotica (2008) volume 26, pp. 679–689. © 2008 Cambridge University Press
doi:10.1017/S0263574708004311 Printed in the United Kingdom

A novel multiple-heuristic approach for singularity-free motion planning of spatial parallel manipulators

Houssem Abdellatif* and Bodo Heimann

Institute of Robotics, Hannover Center of Mechatronics, Appelstr. 11, D-30167 Hannover, Germany

(Received in Final Form: February 7, 2008. First published online: March 27, 2008)

SUMMARY

The issue of motion planning for closed-loop mechanisms, such as parallel manipulators or robots, is still an open question. This paper proposes a novel approach for motion planning of spatial parallel robots. The framework for the geometric modeling is based on the visibility graph methodology. It is opted for a multiple-heuristics approach, where different influences are integrated in a multiplicative way within the heuristic cost function. Since the issue of singularities is a fundamental one for parallel robots, it is emphasized on the avoidance of such configurations. To include singularity-free planning within the heuristic approach, two heuristic functions are proposed, the inverse local dexterity as well as a novel defined "next-singularity" function, in such a way, well conditioned motions can be provided by a single planning procedure. The success of the method is illustrated by some examples.

KEYWORDS: Parallel manipulators; Motion planning; Heuristics; Mechanism singulary.

1. Introduction

Speaking about motion planning of manipulators or robots is speaking about merging two major disciplines of robotics: motion planning on one hand and mechanism science and control on the other hand. The synergy of the methodologies of both sectors has been very successful for serial manipulators and mobile robotics.[1–3] However, path and motion planning for complex closed-loop and parallel mechanisms still remain open questions. To motivate this issue, research efforts in the two categories are briefly reported in the following.

The puristic "path planner" has been looking for general and mathematical optimal solutions for arbitrarily structured mechanisms. For instance, approved methodologies in the range of serial-link mechanisms like probabilistic roadmaps (PRM)[1,2] have also been modified and developed for closed-loop mechanisms with more or less complexity.[4–7] However, most of such approaches aim at the planning for planar linkages,[4] with a few number of closed chains or for cooperating robot arms.[6] Due to the multiple kinematic constraints, the resulting computational effort for finding optimal solutions is huge. Trinkle and Milgram mentioned explicitly about this issue "Planar Test Problems with up to 8 links and two loops were computed in several hours".[8] Consequently they made a return from the PRM paradigm to more mechanism science with consideration of singular configurations in the path planning strategy. Consciously, they considered an obstacle-free workspace and only revolute-jointed mechanisms. They presented very interesting results, which are unfortunately still far from being applicable for parallel manipulators in the practice. The treated mechanisms were mostly planar linkage systems, but a very important contribution of Trinkle and Milgram was the explicit consideration of mechanism's singularities, which is a fundamental issue for parallel manipulators.

The issue of planning remains very rarely treated by "mechanism experts." Mostly, algorithms were proposed for trajectory verification within a hard constrained workspace.[9–11] The paper of Dasgupta and Mruthyunjaya[12] is one of the earliest works that addressed attention to the issue of singularity-free planning for parallel mechanisms. The approach is however restricted to path planning (in contrast to motion planning which includes additionally the presence of obstacles). A further disadvantage is the exclusive geometric character of the approach, which is believed to increase the planning complexity. Recently, an interesting approach was proposed for practical and real parallel manipulators.[13] It shares with our approach of combining and merging multiple methodologies to deliver good and reliable solutions. We think, however, that the proposed cascade of two planning steps is not necessary. Consequently, this paper proposes another general approach that helps planning complete motions, avoiding singularities and obstacles. It is more localized in the second research category and primarily tries to provide planning solutions for parallel mechanisms in a practical way. Our approach is meant to shed some new light on this issue and to excite discussion between "motion planners" and "manipulator operators."

Our approach is characterized by the systematic consideration of all constraints. Besides the configuration-dependent workspace bounds, kinematic coupling, singularities, and obstacles are considered. The latter is especially important for automation and planning in manufacturing cells. Even if some parallel robots have small workspaces, a systematic tool for motion planning is yet required. For that purpose a geometric model of the environment is necessary. The configuration space is chosen to be the six-dimensional pose manifold and not the actuation space, which is for

* Corresponding author. E-mail: abdellatif@ifr.uni-hannover.de, heimann@ifr.uni-hannover.de

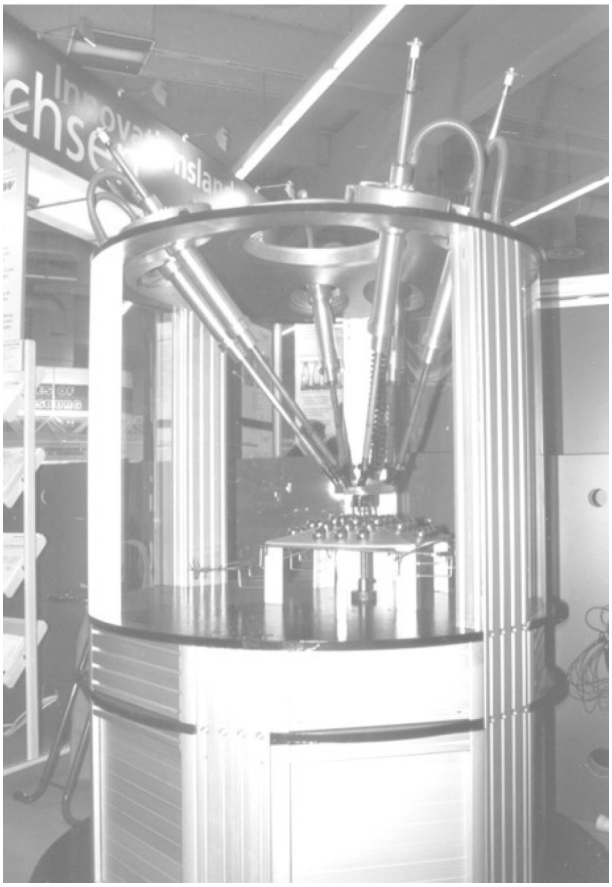

Fig. 1. The case study PaLiDA in a demonstration of planned pick-and-place scenario.

closed-loop mechanisms a variety.[4] This model will be illustrated in Section 2 with a scheme of the classic roadmap method used here for parallel manipulators.

A significant enhancement of performance and robustness of the proposed algorithm is achieved by using multiple heuristics for motion planning and path search. A set of heuristic functions is used for the optimal A*-algorithm. Here, not only the shortest distance to the goal is taken into account but also local dexterity for a well-conditioned motion, as well as the proximity of singularities, are considered. In such manner, a two-step planning, like proposed in ref. [13] become unnecessary. The quantification of the heuristics to account for geometric and kinematic constraints are proposed in Section 3. Singularity avoidance is described in Section 4. The last section presents examples that illustrate the success and effectiveness of the novel proposed approach, which was implemented for the hexapod robot PaLiDA,[14] shown in Fig. 1. The related improvements of planning and computational efficiency are illustrated.

2. Geometric Modeling and Basic Roadmap Method

The basic idea of the planning approach is the geometrical roadmap that is based on the visibility graph (V-Graph) method,[1] which has been extensively studied, especially for 2-D planning problems of wheeled mobile robots. Thus, it is necessary to develop an abstract model for the workcell of the robot. In the following, the case is considered, where the robot's fixed base is mounted above the traveling platform. This is not a limitation to our approach but serves a uniform illustration over the paper. Furthermore, obstacles, objects, and the manipulator itself are modeled as polyhedra (see Figs. 2, 3). For the sake of simplicity and clarity of the illustrations, obstacles are often depicted as prismatic objects.

A preliminary step in the preprocessing is the computation of the six-dimensional obstacle-free workspace $\mathcal{W}_\mathrm{f}$. Classically the kth obstacle $\mathcal{B}_k$ is expanded with an area of forbidden motion, where a collision occurs with the manipulator.[1] For polyhedral objects, computational efficient algorithms have been developed and are state of the art.[15] Nevertheless, it is important to notice the special aspect proposed in this paper, and that is the consideration of the parallel robot also as a polyhedron and not as a linkage system like in refs. [4, 5]. It is believed that this idea is legitimated by the practical aspect in operating parallel manipulators, where the linkage part should not interfere with the operating space. An important handicap for using joint-space variables is the necessity of solving the direct kinematic problem, which is generally not given in a closed form and yields multiple solutions.

The configuration space of the manipulator is defined by a six-dimensional vector $\boldsymbol{x}$ which includes two three-dimensional vectors: the position $\boldsymbol{r}$ of the end-effector with respect to an inertial frame and its orientation $\boldsymbol{\varpi}$, i.e., $\boldsymbol{x}^\mathrm{T} = [\boldsymbol{r}^\mathrm{T}\boldsymbol{\varpi}^\mathrm{T}]^\mathrm{T}$. The latter contains the orientation angles, e.g., as defined by the Roll-Pitch-Yaw convention.[16] The actuator positions are represented by the vector of actuated joints $\boldsymbol{q}_\mathrm{a}$ that can be calculated by the means of the inverse kinematics $\boldsymbol{q}_\mathrm{a} = \boldsymbol{q}_\mathrm{a}(\boldsymbol{x})$.[14]

The construction of the forbidden motion area is done for any obstacle $\mathcal{B}_k$ by expanding it with respect to lth sampled orientation vectors of the end-effector $\boldsymbol{\varpi}_l$ and varying its position around the obstacle. Therefore, expanded obstacles are obtained which are defined with respect to the sampled orientation. This is shown exemplarily in a two-dimensional view in Fig. 2. For each sampled orientation $\boldsymbol{\varpi}_l$ the expanded version $\bar{\mathcal{B}}_k^l$ of an obstacle $\mathcal{B}_k$ is obtained. $\bar{\mathcal{B}}_k^l$ presents the real nonaccessible workspace if the manipulator avoids safely the kth obstacle by keeping a constant orientation $\boldsymbol{\varpi}_l$. The complete inaccessible workspace at any orientation is then given by

$$\bar{\mathcal{B}}_k = \bigcap_l \bar{\mathcal{B}}_k^l. \tag{1}$$

For an obstacle, the workspace that can be attained only at particular orientations is defined as the partially inaccessible workspace

$$\hat{\mathcal{B}}_k = \bigcup_l \bar{\mathcal{B}}_k^l - \bigcap_l \bar{\mathcal{B}}_k^l. \tag{2}$$

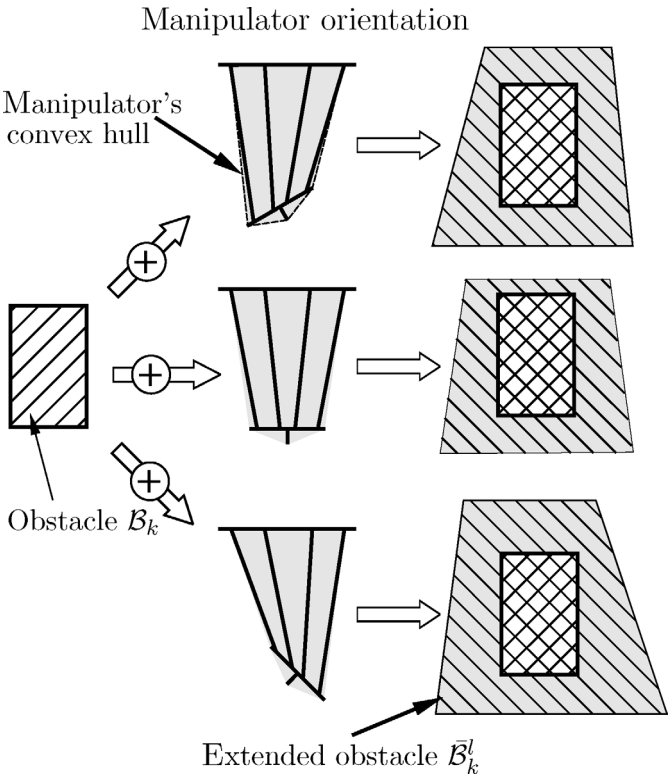

Fig. 2. 2-D view of the expanding procedure of obstacles by taking account the manipulator configuration.

Analogically the completely inaccessible obstacle space results to

$$\bar{\mathcal{B}} = \bigcup_k \bar{\mathcal{B}}_k. \tag{3}$$

The path planning problem is described as finding the path that connects a given starting pose $\boldsymbol{x}_{\mathrm{S}}^{\mathrm{T}} = [\boldsymbol{r}_{\mathrm{S}}^{\mathrm{T}} \boldsymbol{\varpi}_{\mathrm{S}}^{\mathrm{T}}]^{\mathrm{T}}$ with the goal configuration $\boldsymbol{x}_{\mathrm{G}}^{\mathrm{T}} = [\boldsymbol{r}_{\mathrm{G}}^{\mathrm{T}} \boldsymbol{\varpi}_{\mathrm{G}}^{\mathrm{T}}]^{\mathrm{T}}$ and that remains included in the partially available space $\mathcal{W}_{\mathrm{f}} = \mathcal{W} - \bar{\mathcal{B}}$.

The next step is then to construct a visibility graph-based roadmap that presents feasible connections between the start and goal configurations. This is achieved by first defining what we call a S-G line (start to goal line), which connects the start position $\boldsymbol{r}_{\mathrm{S}}$ with the goal position $\boldsymbol{r}_{\mathrm{G}}$. Let us consider a nontrivial case, i.e., the S-G line passes through the obstacle space $\bar{\mathcal{B}}$. Thus, an obstacle-free path is required. Let $\mathcal{P}_{\mathrm{SG}}$ be the set of planes that contain the S-G line and $\mathcal{P}_{\mathrm{SG}}^m$ be the mth plane defined by the rotation of an angle α_m. In combination with the constant S-G line only α_m is necessary for the exact definition of a corresponding S-G plane $\mathcal{P}_{\mathrm{SG}}^m$ of the set.

The construction procedure begins by considering a first sample $l = 1$ of the manipulator orientation $\boldsymbol{\varpi}_1$ that yields a corresponding inaccessible polyhedral space $\bar{\mathcal{B}}^1$. Afterwards the set of S-G plane is considered by varying the sample orientation α_m and computing the intersection of each resulting plane $\mathcal{P}_{\mathrm{SG}}^m$ with the vertices of $\bar{\mathcal{B}}^1$. The result is sets N_m^1 of discrete three-dimensional positions in the workspace, which contain possible connections from the start to the goal, for the case when the manipulator keeps a constant orientation $\boldsymbol{\varpi}_1$. An exemplarily starting situation

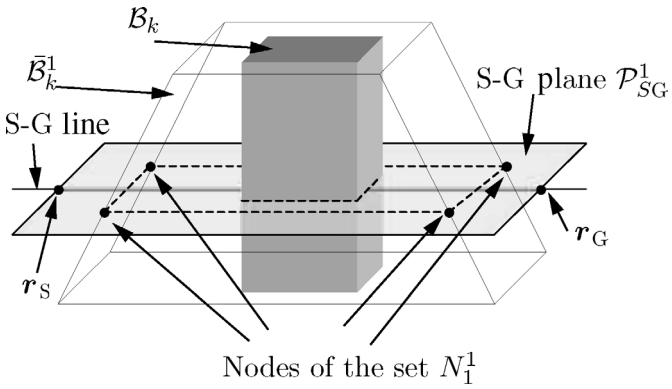

Fig. 3. Basic idea for constructing geometric roadmap by the presence of obstacles. Example given for the starting situation with $l = 1$ and $m = 1$.

is illustrated in Fig. 3. More generally and for a sampled orientation $\boldsymbol{\varpi}_l$ a set of corresponding feasible connections is obtained by the same procedure

$$N^l = \bigcup_m \underbrace{\left(\mathcal{P}_{\mathrm{SG}}^m \cap \mathcal{V}(\bar{\mathcal{B}}^l)\right)}_{N_m^l} \subset \mathbb{R}^3, \tag{4}$$

where $\mathcal{V}(\cdot)$ denotes the vertices of a polyhedron. The corresponding graph nodes in the six-dimensional configuration space is obtained by extending the found connections by the related sampled manipulator orientation

$$\mathcal{N}^l = \{(N^l, \boldsymbol{\varpi}_l)\} \subset \mathbb{R}^6. \tag{5}$$

Now repeating the described procedure for different $\boldsymbol{\varpi}_l \in [\boldsymbol{\varpi}_{\min}\ \boldsymbol{\varpi}_{\max}]$ yields the complete graph

$$\mathcal{N} = \bigcup_l \mathcal{N}^l. \tag{6}$$

The procedure of building the V-Graph is given by Algorithm 1 and depicted in Fig. 4. For the sake of clarity, a prismatic obstacle is considered for two sampled end-effector orientations, and only most relevant connections are shown. The final connected spatial visibility graph can be imagined as depicted in Fig. 5.

Discussion: Since the manipulator is modeled by different convex hulls, depending on the regarded orientation, an important set of $\mathcal{N}$ do lay in the partially accessible workspace $\hat{\mathcal{B}}$ (see Eq. (2) for definition). Such positions are accessible only at a particular orientation of the end-effector. This issue is treated by defining the graph to be the set of poses rather than positions (5). This will of course increase the necessary computational effort, since the planning is required in a six-dimensional space. However, such concept will provide a powerful planning approach, because the partially accessible volume is used for planning. If the manipulator is roughly modeled by one single hull for all relevant orientations, the planning can occur in the three-dimensional workspace. Such an approach is however very conservative, since the partially accessible workspace

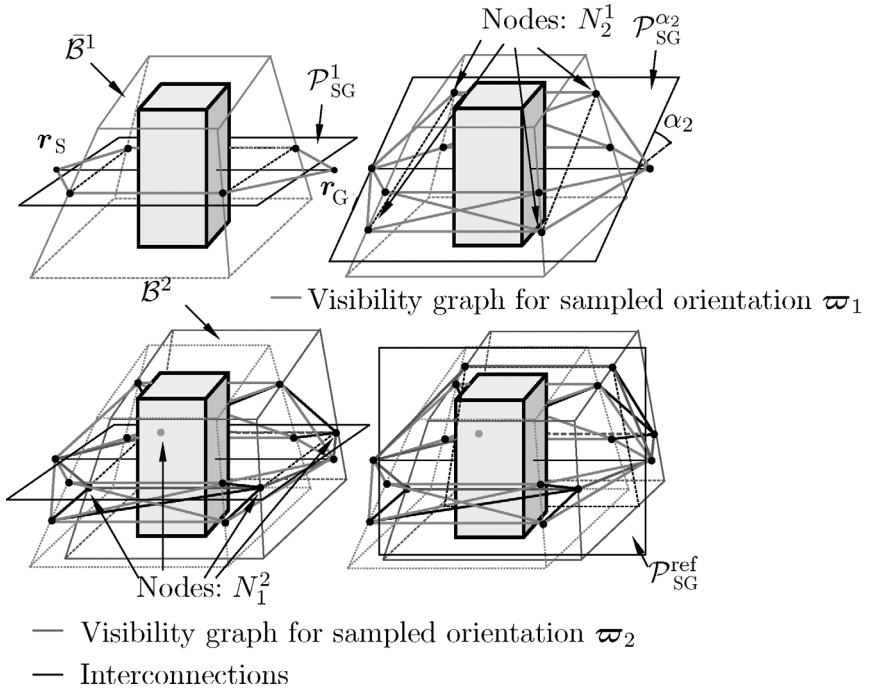

Fig. 4. Constructing geometric visibility graph-based roadmap for the case of one obstacle.

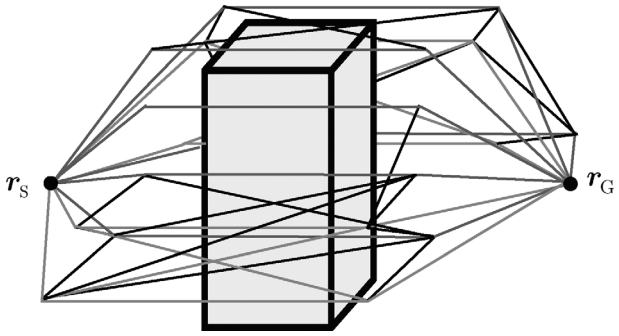

Fig. 5. Spatial visibility graph for the case of one obstacle.

is wasted. The planner may fail to find a path, even when one exists. The discussed issue is highly relevant and important for parallel manipulators. The completely accessible workspace at all orientations is a very small part of the maximal workspace and is not sufficient for suitable planning.

Even for simple environment structures and low number of obstacles the obtained graphs are highly dense and complex. The complexity depends on the sampling density of both manipulator orientation $\boldsymbol{\varpi}_l$ and the orientation α_m of the S-G planes $\mathcal{P}_{SG}^m$. Exemplarily for a single obstacle, the complexity of the respective roadmap depends in $\mathcal{O}(n)$ on the sampling density of α_m and in $\mathcal{O}(n^2)$ on that of $\boldsymbol{\varpi}_l$. Here, it is recommended to determine appropriate densities to achieve a tradeoff between accuracy and computational effort. In our work and for the studied system we sampled α_m in intervals of 10° between −90° and +80°. The manipulator's orientation was sampled by 5° steps for the complete range of possible orientations. With appropriate search techniques, it is possible to achieve time-efficient planning (see the following sections).

Algorithm 1. Algorithm for building up a spatial visibility graph

Initialization:
- Let $\mathcal{N} = \emptyset$.
- Build Start-Goal Line: $\mathcal{L} = (\boldsymbol{r}_S(\boldsymbol{x}_S)\, \boldsymbol{r}_G(\boldsymbol{x}_G))$.

Computation:
for $\boldsymbol{\varpi}_l \in \{\boldsymbol{\varpi} \in [\boldsymbol{\varpi}_{min}\ \boldsymbol{\varpi}_{max}]\} \subset \mathbb{R}^3$ **do**
  consider $\bar{\mathcal{B}}^l = \bigcup_k \bar{\mathcal{B}}_k^l$ and its vertices $\mathcal{V}^l$
  for $\alpha_m \in \{\alpha \in [\alpha_{min}\ \alpha_{max}]\} \subset \mathbb{R}$ **do**
    Build $\mathcal{P}_{SG}^m = \text{plane}\{\mathcal{L}\, \alpha_m\}$
    Compute the intersection $N_m^l = \mathcal{P}_{SG}^m \cap \mathcal{V}^l \subset \mathbb{R}^3$
  end for
  $\mathcal{N} = \mathcal{N} \cup \left(N_m^l, \boldsymbol{\varpi}_l\right) \subset \mathbb{R}^6$
end for
connect($\mathcal{N}$, $\boldsymbol{x}_S$, $\boldsymbol{x}_G$)

3. Multiple Heuristics Approach

For the proposed geometric model and roadmap approach, heuristic search techniques are highly appropriate. Especially powerful modified A*-algorithms improve the planning in terms of the final solution's quality and the computational cost. The improvement of the classic A*-algorithm depends on the defined and chosen heuristic functions, which has been studied for motion planning of serial manipulators.[3,17,18] In our work we found out that multiple heuristics improve the planning of parallel manipulators. If only the classic distance-to-goal heuristic function is considered, the graph search occurs rather in the breadth and demands more computational time. This is due to the compactness of the available workspace, such that significant change of the distance-to-goal values are minor. Furthermore, the most important advantage of using multiple heuristics is the simple and efficient consideration of singularities, well

conditioning and even more mechanism properties within the path planning. Complex arithmetic problems like those discussed in refs. [8, 12, 19] are not necessary any more, since neither the computation of constraints nor the calculation of direct kinematics has to be achieved. Of course, this property is also due to the definition of planning task in the operational workspace.

The search strategy is based on minimizing the evaluation function $f(n) = g(n) + h(n)$. Here, $g(n)$ is the cost from the starting node $\boldsymbol{x}_S$ to the actual node $\mathcal{N}(n)$. The function h is the heuristic function and is the estimated cost up to the final goal $\boldsymbol{x}_G$ and characterizes the search strategy. It may include multiple heuristic costs (also called influences) denoted by c_k. Many authors proposed the linear combination of these subheuristic functions[17,18]

$$h(n) = \sum_k w_k \, c_k(n). \tag{7}$$

Such linear combination can be counterproductive, since the scale of the subheuristics can interfere or neutralize each other.[3] The tuning of the weights w_k can help to overcome this drawback, but it demands considerable *a priori* knowledge or high amount of trial-and-error steps, especially in the case of a big number of multiple subheuristics (in general >3). To avoid the scaling problem, it is proposed to combine c_k by multiplication:

$$h(n) = W \prod_k c_k(n), \tag{8}$$

where W is a constant. Notice that by using a multiplication approach, the individual scaling weights w_k become meaningless. It is proposed to use a single weight W only for numerical reasons to avoid, e.g., extensively small or extensively high heuristic function. The condition is set that $c_k(n)$ can be equal to zero only at the goal configuration

$$\forall k, \; \exists \, c_k(n) = 0 \iff n = \arg(\mathcal{N}(n) = \boldsymbol{x}_G). \tag{9}$$

For the case of spatial parallel manipulators two classes of subheuristics are proposed: primary and secondary ones.

3.1. Primary heuristics

The primary heuristics design the influences or functions that are necessary for planning a geometrical and singularity-free discrete sequence of nodes from the start pose to the goal pose. These heuristics are necessary to find optimal solutions in the sense of A*- or modified A*-search techniques. Following influences are proposed as primary:

- *Distance to the goal position* $\boldsymbol{r}_G$ is the most classic heuristics type and can be simply computed by

$$c_1(n) = d_r(n) = \|\boldsymbol{r}(n) - \boldsymbol{r}_G\|_2. \tag{10}$$

  The distance is chosen rather than pose difference, because it helps decreasing the computational cost. Furthermore, pose difference depends on the used orientation formalism and has no physical meaning.

- *Changes in actuator position* is defined as

$$c_2(n) = \Delta q(n) = \|\boldsymbol{q}_a(n) - \boldsymbol{q}_a(\boldsymbol{x}_G)\|_2. \tag{11}$$

  In analogy to the distance to the goal, this influence can be used as a single heuristic function for complete path planning. In such cases however, accurate solutions of the direct kinematics must be provided.

- *Inverse dexterity* is the inverse of the mechanism's local dexterity index defined by

$$c_3(n) = \kappa^{-1}(n) = \text{cond}(\boldsymbol{J}^{-1}(\boldsymbol{x}(n))), \tag{12}$$

  where the inverse Jacobian of the mechanism is defined as $\boldsymbol{J}^{-1} = \frac{d\dot{\boldsymbol{q}}_a}{d\dot{\boldsymbol{x}}}$. Minimizing this heuristic influence is equivalent to choosing the path with better conditioning and better kinematic precision.[12, 16] Since singular configurations correspond to infinite $\kappa^{-1}(n)$,[16] using the proposed heuristic search will avoid such configurations automatically. In contrast to other approaches proposed in refs. [8, 12, 13] there is absolutely no need of complex geometric calculations, just the numerical evaluation of κ^{-1} is sufficient for planning singularity-free pose sequences. This idea is one of the major contributions of this work. The condition number should be regarded as a relative indicator between different configurations. Its use for planning purpose is justified since such procedure occurs with respect to a constant reference frame and for a specified manipulator, with well-defined geometry.

3.2. Secondary heuristics

The secondary heuristics are not supposed to be less important. They are called secondary because they contribute mostly to the increasing of the computational performance and to the fine-tuning of the final motion that occurs between the discrete poses. They can integrate properties of the manipulator and preferences of the operator.

- *Distance to a reference S-G plane* allows to choose a favored planning direction or a reference S-G plane $\mathcal{P}_{SG}^{ref}$:

$$c_4(n) = d_\perp(n) = d\big(\boldsymbol{x}(n), \mathcal{P}_{SG}^{ref}\big). \tag{13}$$

  Motions far from the favored plane are penalized. The choice of such plane depends on the manipulator's working configuration. It is recommended to choose the one that provides wide intersection with the workspace. In this work the vertical plane is favored (see example in Fig. 4 at the bottom right).

- *Orientation change*: The roadmap provides nodes with different orientations of the end-effector. Connecting two nodes with different orientations is critical while surpassing obstacles, since collisions of the end-effector with obstacles may occur. In such case, a collision check has to be performed while (or after) the planning. To avoid such scenario orientation change

$$c_5(n) = \Delta\varpi = \sum_{i=1}^{3} |\varpi_i(n) - \varpi_i(\boldsymbol{x}_G)| \tag{14}$$

can be penalized. The planning algorithm is enforced to choose solutions, where the slope of the end-effector is kept constant while surpassing obstacles.

- *Distance to the next singular location* is defined by the location of the next singularity situated in a specific direction. It penalizes graph connections with possible singularities in between (see Section 4.2 for the mathematic definition). This influence helps avoiding traveling toward near singular configurations or workspace boundaries. It simplifies and improves the avoidance of concave regions, holes, or singularity bulks. Trajectory verifications like those proposed in refs. [9, 20] become unnecessary.

The following section focuses on the singularity avoidance and on the related proposed heuristics.

4. Singularity Avoidance in Path Planning

The issue of singularities is decisive for parallel manipulators, since they provoke the degeneration of controllability and may damage the mechanism. For that reason, this section is dedicated to the singularity avoidance in motion planning. To the best knowledge of the authors, only two works have dealt previously with this problem for parallel manipulators.[12,19] Even if the proposed methods are different, both have a two-step strategy in common. The first step consists in providing a nominal path, which should be as much as possible free from singularities. The second step is to check and avoid the rest of the singularities if the first avoidance step failed some how. In ref. [12] a divide-and-conquer strategy was proposed, which requires an undefined, extensive, and important amount of trial-and-error operations. In refs. [13, 19], singularities are avoided in the second step by using Grassmann geometry.[16]

The drawbacks of two-step strategies are eliminated by using the proposed multiple-heuristics approach. Two functions c_3 and c_6 are used. The condition of the inverse Jacobian is used to avoid choosing nodes or poses that are near or in singularities, where $c_3(n) \gg 0$. Since our geometric model is not based on cell decomposition but on the computational efficient visibility graphs, the space between two successive nodes remain unchecked, if only c_3-heuristics are considered. For that reason, c_6-heuristics are used to avoid traveling toward a near singularity. For study of singularities, the great research works of Merlet, Gosselin, Bonev[21] and others can be referred to. For planning purpose the location of all singularities can be determined by numerical validation over the workspace.[16] In the following, it is not differentiated between singularities of Type I, II, and III as defined in.[22] All have to be avoided using and according to the same heuristic approach.

4.1. c_3-Heuristics: Inverse dexterity

Bad conditioned regions of the workspace are characterized by a high condition number of the inverse Jacobian. They correspond to small local dexterity index. In the sense of the A*-algorithm, where a cost function has to be minimized, the inverse of the dexterity index is introduced as a heuristic cost or influence. Its minimization while planning will lead to the increasing of the dexterity and therefore to the bettering of the conditioning along the planned path. Figure 6 gives an illustrative example of how the inverse dexterity is distributed in the workspace of parallel manipulators. It is a case study of our hexapod prototype PaLiDA.[14,23] In singularities, such as the border of the workspace $\kappa^{-1} = (\mathrm{cond}(\boldsymbol{J}^{-1}))^{-1}$ becomes infinite. It means, if we take the realistic case of having the starting and goal poses within the workspace, using the inverse dexterity as a heuristic influence will never provide path nodes that are outside the workspace or even close to its border or any singularities, since this would mean the increasing of the cost function h. Furthermore, the algorithm will find always a feasible path, as long as there is a real and obstacle-free connection between the start and goal within the workspace and as long as the resolution of the roadmap is high enough. This property of resolution complete has been proven to be the case of general PRM methods.[1,2] A successful planning is achieved independently from the geometric form and complexity of the workspace that could have concave surfaces (like the examples given in refs. [12, 20]) or include singularities. Of course, the success of planning is not only due to the c_3-heuristics but also to its combination with the classic distance-to-goal c_1-heuristics.

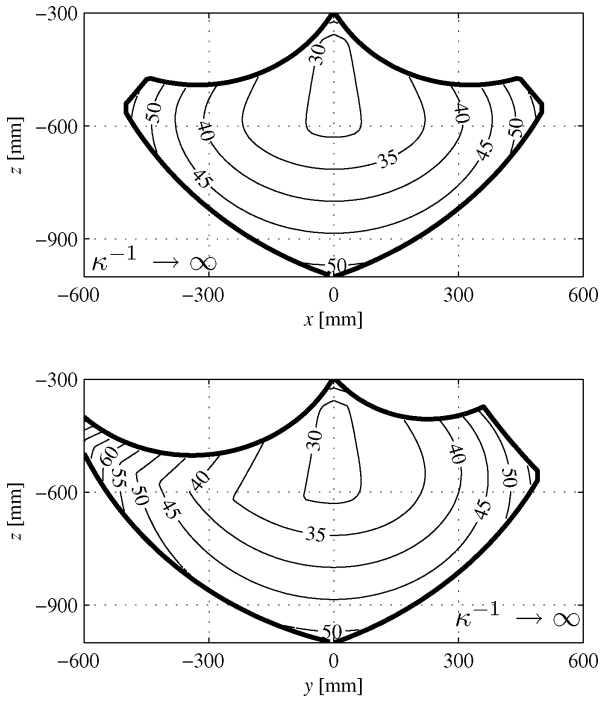

Fig. 6. Isolines of the inverse dexterity within 0-orientation workspace (top: xz-plane, bottom: yz-plane).

4.2. c_6-Heuristics: The next singularity function

The output of the planning algorithm should be a set of singularity-free configurations. The manipulator will move from one to the next. To guarantee that no singularities exist between the planned nodes, c_6-heuristic is introduced in the multiple-heuristics search, to keep the all-in-one-step approach of our planning algorithm. First some definitions are introduced.

Let $\mathcal{S}$ be the set of all singular configurations $\boldsymbol{x}_\sigma$ and let $\mathcal{S}_\mathrm{L}$ be the set of the corresponding locations.

$$\mathcal{S}_L = \left\{\boldsymbol{r}_\sigma(\boldsymbol{\varpi}) = [x_\sigma \; y_\sigma \; z_\sigma]^\mathrm{T}, \; \boldsymbol{x}_\sigma = \left[\boldsymbol{r}_\sigma^\mathrm{T} \; \boldsymbol{\varpi}_\sigma^\mathrm{T}\right]^\mathrm{T} \in \mathcal{S}\right\}. \quad (15)$$

We consider now the vector connecting an arbitrary position $\boldsymbol{r}$ in the workspace with a singularity location

$$\boldsymbol{a}_\sigma(\boldsymbol{\varpi}) = \boldsymbol{r}_\sigma(\boldsymbol{\varpi}) - \boldsymbol{r}. \quad (16)$$

The basic idea is to transform such vectors in spherical coordinates

$$\boldsymbol{a}_\sigma(\boldsymbol{\varpi}) \; \longrightarrow \boldsymbol{\xi}_\sigma(\boldsymbol{\varpi}, \boldsymbol{r}) = [\varphi_\sigma \; \theta_\sigma \; \rho_\sigma]^\mathrm{T}, \quad (17)$$

where

$$\begin{aligned} \rho_\sigma &= \sqrt{(x_\sigma - x)^2 + (y_\sigma - y)^2 + (z_\sigma - z)^2} \\ \varphi_\sigma &= \arctan\left(\frac{y_\sigma - y}{x_\sigma - y}\right) \\ \theta_\sigma &= \arctan\left(\frac{\sqrt{(x_\sigma - x)^2 + (y_\sigma - y)^2}}{(z_\sigma - z)^2}\right) \end{aligned} . \quad (18)$$

such that for every position $\boldsymbol{r}$ the proximity of singularities in the spherical directions φ and θ can be defined by the value ρ_σ. Based on this we define the "next-singularity function":

$$\rho_\sigma = f_\sigma(\boldsymbol{r}, \boldsymbol{\varpi}, \varphi, \theta). \quad (19)$$

In other words f_σ gives the euclidian distance ρ_σ to the next singularity location if the platform travels from the position $\boldsymbol{r}$ with the orientation $\boldsymbol{\varpi}$ in the spherical directions φ and θ. The proposed approach is necessary because spatial parallel mechanisms have mixed types of degrees of freedom (translations and rotations). It is therefore not possible to define a mathematical definition for the distance to a singular configuration. Consequently, the distance is defined with respect to the translational dof's and for given arbitrary orientation. A visualization of the proposed spherical function is given in Fig. 7. The integration of such formalism into the c_6-heuristics is quite simple. It is supposed to calculate the cost related to c_6-influence between an actual graph node $N_1(\boldsymbol{r}_1, \boldsymbol{\varpi}_1)$ and a next one $N_2(\boldsymbol{r}_2, \boldsymbol{\varpi}_2)$. In the same manner as proposed for the singularity locations, the connection between the two nodes $\boldsymbol{r}_2^1 = \boldsymbol{r}_2 - \boldsymbol{r}_1$ is transformed in spherical coordinates

$$\boldsymbol{r}_2^1 \; \longrightarrow \boldsymbol{\xi}_2 = [\varphi_2 \; \theta_2 \; \rho_2]^\mathrm{T}, \quad (20)$$

to obtain the necessary spherical directions φ_2 and θ_2, and the related distance ρ_2 to travel from N_1 to N_2. The next singularity function (19) is then validated for φ_2 and θ_2 to calculate $\rho_\sigma(\varphi_2, \theta_2) = f_\sigma(\boldsymbol{r}_1, \boldsymbol{\varpi}, \varphi_2, \theta_2) \triangleq \rho_\sigma(N_1 \rightarrow N_2)$. It is obvious that traveling from one node to the other could cross a singularity or workspace boundary, if $\rho_\sigma(\varphi_2, \theta_2) \leq \rho_2 \triangleq \rho(N_1 \rightarrow N_2)$. In such case, the connection between the two nodes has to be forbidden by attributing an infinite high

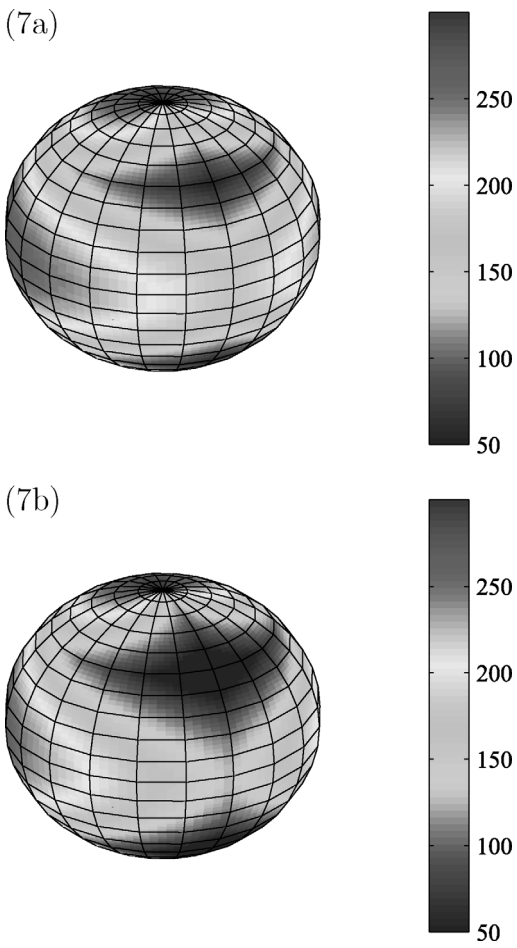

Fig. 7. An example of the spherical next-singularity function: (7a) for $\boldsymbol{x} = [0.0, -0.3, -0.6, 0.0, 0.0, 0.0]$; (7b) for $\boldsymbol{x} = [0.0, -0.3, -0.6, \pi/6, 0.0, 0.0]$.

cost:

$$\begin{aligned} c_6(N_1 \rightarrow N_2) &= \infty \quad \text{if } \rho_\sigma(N_1 \rightarrow N_2) \leq \rho(N_1 \rightarrow N_2) \\ c_6(N_1 \rightarrow N_2) &= 1 \quad \text{if } \rho_\sigma(N_1 \rightarrow N_2) > \rho(N_1 \rightarrow N_2) \end{aligned} . \quad (21)$$

With such measure, it is impossible for the algorithm to choose directions or connections that could lead to the worst case of intersecting a singularity, singularity bulks or that could partially leave the workspace. The sampling nature of PRM plays here an important role, since it provides the search algorithm with alternative connections. A simple example is given in Fig. 8. To travel form $\boldsymbol{r}_\mathrm{S}$ to $\boldsymbol{r}_\mathrm{G}$, four nodes are possible as intermediates. Certainly, node N_1 is the most advantageous in terms of the classic c_1-heuristics. Without regarding c_6-influence it is possible to plan the sequence $\{\boldsymbol{r}_\mathrm{S} \rightarrow N_1 \rightarrow \boldsymbol{r}_\mathrm{G}\}$, which would result in a path that crosses the singularities at workspace boundaries. By introducing c_6 the cost of traveling to N_1 or to N_2 becomes infinite, which yields the rejection of these nodes and planning the sequence $\{\boldsymbol{x}_\mathrm{S} \rightarrow N_3 \rightarrow \boldsymbol{r}_\mathrm{G}\}$. The possible connection via N_4 is rejected due to the c_1-heuristics, since traveling through N_4 increases the corresponding cost and therefore destination to the goal. Once again, the proposed method is able to recognize feasible and reliable connections by simple evaluation of heuristic functions. Two further remarks have to

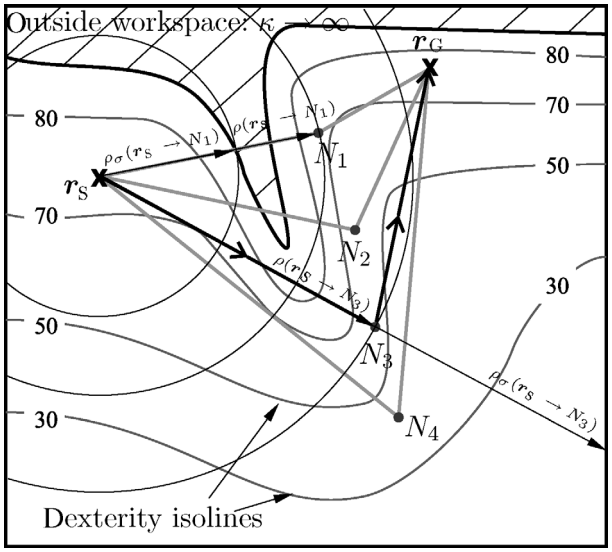

Fig. 8. Example of obstacle avoidance planning using inverse dexterity for well-conditioned and singularity-free paths.

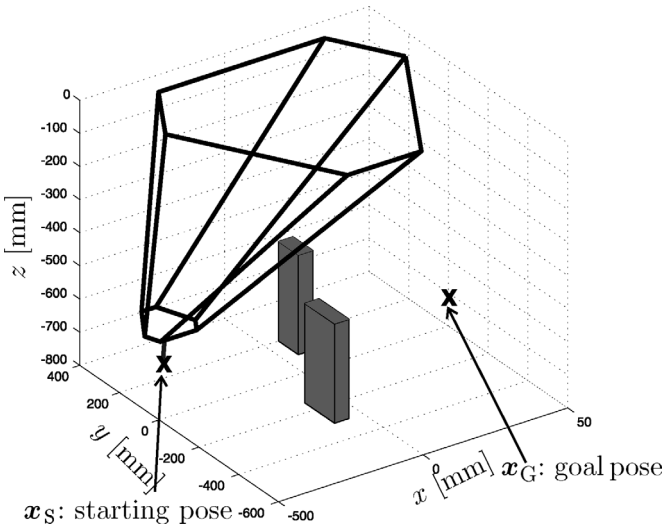

Fig. 9. Planning example for the hexapod PaLiDA, initial position and situation with two obstacles.

be considered for practical implementation. First, the infinite cost in Eq. (21) can be programmed as a very high constant. Second, Eqs. (15–19) do not have to be evaluated or known *a priori* for all workspace configurations. It is sufficient to evaluate the equations just for the graph nodes considered by the searching algorithm.

5. Examples and Results

The planning approach was integrated in a software concept running on a conventional PC with a Windows operating system. The interaction with the user is achieved under MATLAB via graphical user interfaces (GUIs). In a main interface, the user can define or import geometric data of the considered spatial manipulator. Furthermore, starting and goal pose can be given along with the desired heuristic functions to be evaluated. Additional GUIs are used for the design of the workcell with desired features or obstacles or for the visualization of the final planned motion. In the near future, import tools should be available to integrate CAD models into the software. The computational geometry algorithms that concern the preprocessing and the geometric model described in Section 2 are written in C/C++ and connected to the main software. In the following, the proposed planning approach is demonstrated with an example. Here, the considered manipulator corresponds to the real system PaLiDA,[23] which is supposed to move from pose $\boldsymbol{x}_S = [-0.35\,\text{m},\ 0.2\,\text{m},\ -0.75\,\text{m},\ 7^\circ,\ 0^\circ,\ 0^\circ]$ to pose $\boldsymbol{x}_G = [0.3\,\text{m},\ -0.3\,\text{m},\ -0.66\,\text{m},\ 0^\circ,\ -5^\circ,\ 0^\circ]$ avoiding two obstacles, as depicted in Fig. 9. The situation is such that the two obstacles are separated with a narrow corridor. The manipulator can pass by only if it moves upwards or if it goes around the two obstacles to attain the desired goal.

The planning algorithm was executed for the same scenario with different modes, varying each time the number of the considered heuristic functions. The six different types of influence adjustments are summarized in Table I. We shall call the different planning trials as planning modes, which are numerated from 1 to 6.

The results for the six planned paths are very interesting and comply very well with the multiple-heuristics idea behind our concept. The planning results are given in detail in Table I and are shown in Fig. 10. These results are given primarily in the form of node sets that contain the sequence of the planned six-dimensional poses. This is the direct output of the planner and is depicted in Figs. 10 and 11 as circles. The executable motion can be obtained by interpolating a path between the poses. Here many approaches can be used, such as simple linear interpolation, B-splines, and cubic splines. However, interpolation may provoke additional uncertainties in the geometric model, such that the interpolated path can violate forbidden regions of motions. This can be though simply regarded during the preprocessing (e.g., while expanding objects), if the interpolation method is known in advance, which is actually always the case.

The planning *mode1* is the closest mode to the classic A* -single heuristic approach. It yields a very instinctive solution, where the robot moves toward the goal pose while lifting the platform to avoid hurting the obstacle with the tip of the tool. While surpassing the obstacles, the robot inclines its end-effector platform by $\boldsymbol{\varpi} = [-10^\circ\ \pm 20^\circ\ 0^\circ]$. With horizontal platform, the robot must move further upwards, which will increase the heuristic cost. Critical is that the end-effector changes its orientation while surpassing the obstacle. This explains why it surpasses the obstacles at different entry and exit levels of $-0.411\,\text{m}$ and $-0.400\,\text{m}$, respectively. To avoid such critical solutions the c_5-heuristics are considered, which lead to the solution given by *mode2*. Here the robot maintains its end-effector orientation while surpassing both obstacles, but it has to move further upwards ($z = -0.387\,\text{m}$) to achieve that. The computation time is larger than in the case of *mode1*. The impact of the c_4-heuristics on the computational time is noticeable for the cases when a few primary heuristics are used. For the scenario studied here, running *mode1* without considering the distance to the reference plane increases the computational time

Table I. Planning example: Adjustments of the planning modes with the corresponding results of planning, computation time, and selected path nodes (or poses).

Modes	c_1	c_2	c_3	c_4	c_5	c_6	Time (s)	Nodes
mode1	1	–	–	1	–	1	63.46	−0.050 −0.002 −0.411 −10° 20° 0° 0.043 −0.074 −0.400 −10° −20° 0°
mode2	1	–	–	1	1	1	70.81	0.004 −0.083 −0.387 −20° −10° 0° 0.038 −0.110 −0.387 −20° −10° 0°
mode3	1	1	–	1	1	1	27.98	−0.086 0.345 −0.451 10° 0° 0° −0.010 0.266 −0.378 10° 0° 0° 0.189 0.068 −0.378 10° 0° 0° 0.253 −0.028 −0.449 10° 0° 0°
mode4	1	1	1	1	1	1	21.41	−0.086 0.343 −0.452 0° 0° 0° −0.012 0.266 −0.380 0° 0° 0° 0.189 0.065 −0.380 0° 0° 0° 0.251 −0.026 −0.449 0° 0° 0°
mode5	1	–	1	1	1	1	67.85	0.051 0.001 −0.396 0° 0° 0° 0.048 −0.076 −0.392 0° 0° 0°
mode6	–	1	1	1	1	1	66.90	−0.381 0.019 −0.595 0° 0° 0° −0.225 −0.439 −0.422 0° 0° 0° 0.082 −0.474 −0.450 0° 0° 0°

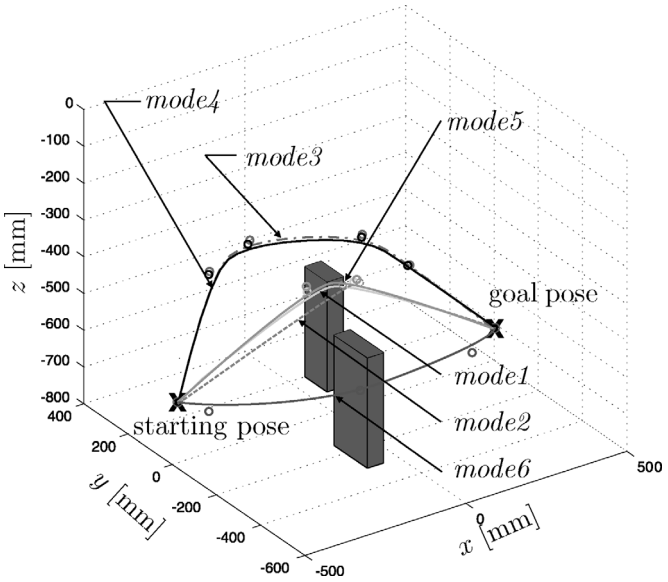

Fig. 10. Planning example for the hexapod PaLiDA, initial position with two obstacles.

of about five seconds. Running *mode4*, *mode5*, or *mode6* without c_4-heuristics does not lead to a significant change of the computational time. It shall be noticed that the effect of c_6-heuristics on the computational time is minor but is crucial for avoiding singularities, as shown in the planning example in Fig. 8.

By introducing the c_2-heuristics, the computation time is reduced significantly, as shown by *mode3*. The c_2-influence is proposed for penalizing the changes of actuator lengths. Its consideration has led to a solution where the robot tends to go around the obstacles, rather than surpassing them from above. The computation time is improved but the found path is longer and more complicated. The user or robot operator may judge, which is the preferable solution, by comparing, e.g., the results of the dynamic time-optimal planning, as we proposed in a previous paper.[23] By additionally including c_3-heuristics, a further improvement of computation time is clearly noticed. *mode4* yields apparently a similar solution as the previous case. This mode however corrects the orientation of the end-effector to zero, while the path geometry remains almost equal. Sloping the platform of the parallel manipulator reduces the dexterity and manipulability of the mechanism.[16] To cope with this inconvenience, c_3-heuristics helps correcting the orientation and provides a better conditioned (and implicitly also singularity-free) motion. This can be also observed with *mode5*. Here c_2-influence is not regarded to compare *mode5* and *mode2*. By considering the inverse dexterity as a cost factor, the orientation of the platform is corrected to be 0°, while avoiding the two obstacles. As it can be noticed from *mode6*, it is not the c_2-heuristics alone that are responsible for improving the computational efficiency but the combination with other influences. In *mode6* the primary heuristics c_1 were not regarded. As a result, the end-effector moves around the obstacles at approximately the same height, which results in a longer path than those obtained by all other modes. Additionally, the necessary computational time increases. We can conclude that the single influences are responsible for the geometry of the planned path, whereas the improvement of the computational time is due to their combination in a multiplicative way.

Generally, the planning solutions are very good and also very sensible to the choice of heuristics. They may not be absolutely optimal, but they are obtained in a computational

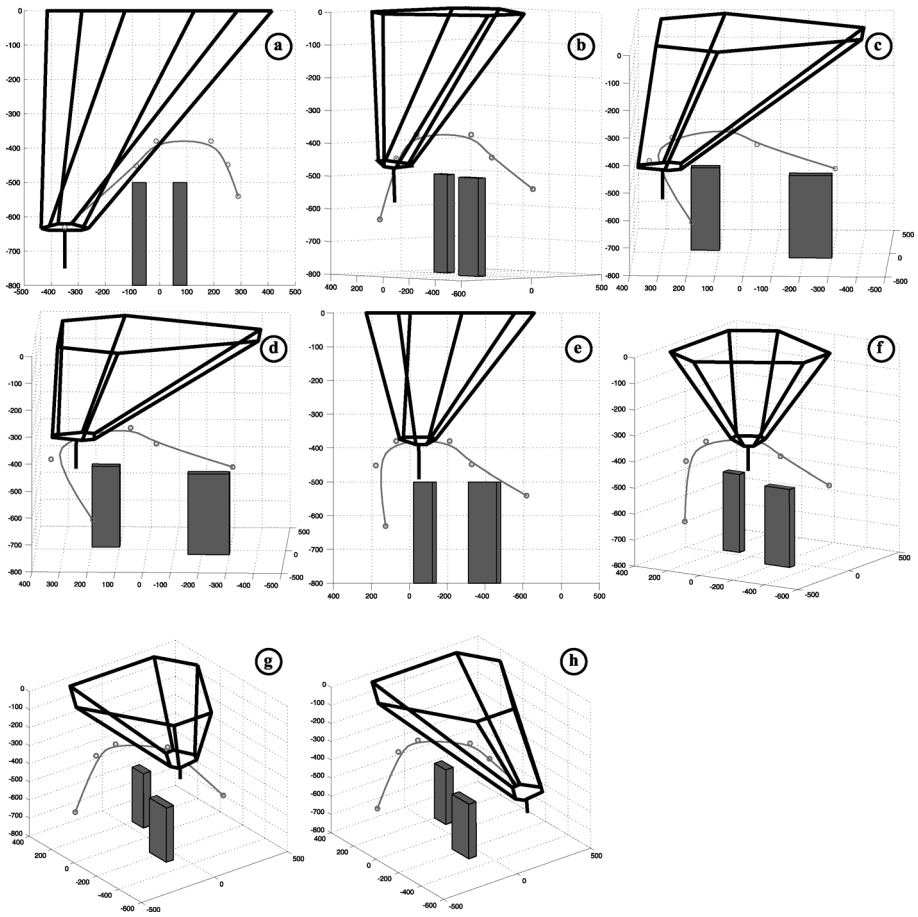

Fig. 11. Sequence of motion along the planned path with *mode4*. The captures are shown from different perspectives.

efficient manner. The solutions are comprehensible and very close to the practical reality. Figure 11 shows the solution of *mode4* as motion sequence given in different perspectives.

6. Conclusions and Final Remarks

In this paper a novel multiple-heuristics approach was presented for planning singularity- and obstacle-free motions for spatial parallel manipulators. The central idea was to give a practical approach with a one-step solution that copes with most of the challenging issues concerning planning for closed kinematic structures. The methodology is based on a six-dimensional roadmap geometric model of the environment. Visibility graph techniques have been used for building up the search graph. The heuristic search has been enhanced by introducing multiple influences. The multiplication of the associated costs yields improvement of computational time. By introducing new ideas for such influences, like the inverse dexterity of the mechanism or the next singularity function, well-conditioned and singularity-free motions are provided. The solution is obtained by a one-step planning such that no additional verification is necessary. The approach has been demonstrated by an exemplarily chosen scenario with a parallel mechanism that corresponds to a real system.

The aim of this paper was also to provide the discussion of planning closed-loop mechanisms with a new idea. The presented solutions are optimal only in the heuristic sense and are not absolutely optimal. The geometric modeling of the manipulator itself was rather rough as a polyhedron and not as a linkage system. Nevertheless, heuristic optimal motion planning is achieved in a computationally efficient manner.

References

1. J.-C. Latombe, *Robot Motion Planning* (Kluwer Academic Publishers, Boston, 1991).
2. T. Siméon, J. P. Laumond, J. Cortes and A. Sahbani, "Manipulation planning with probabilistic roadmaps," *Int. J. Robotics Res.* **23**(7), 729–746 (2004).
3. D. Mages, B. Hein and H. Wörn, "Introduction of additional information to heuristic path planning algorithms," **In**: *Proceedings of the International Conference on Mechatronics and Robotics 2004*, MechRob2004, Aachen, Germany (2004) pp. 1028–1033.
4. S. M. LaValle, J. H. Yakey and L. E. Kavraki, "A probabilistic roadmap approach for systems with closed kinematic chains," **In** *Proceedings of the 1999 IEEE International Conference on Robotics and Automation*, Detroit, MI (1999) pp. 1671–1676.
5. J. H. Yakey, S. M. LaValle and L. E. Kavraki, "Randomized planning for linkages with closed kinematic chains," *IEEE Trans. Robotics Automation* **17**(6), 951–958 (2001).
6. J. Cortes and T. Simeon, "Sampling-based motion planning under kinematic loop closure constraints," **In** *Proceedings of the Workshop on the Algoritmic Foundations of Robotics*, WAFR 2004, Utrecht, the Netherlands (2004) pp. 59–74.
7. O. B. Bayazit, D. Xie and N. M. Amato, "Iterative relaxation of constraints: A framework for improving automated motion planning," **In** *Proceedings of the 2005 IEEE/RSJ International Conference on Intelligent Robots and Systems*, IROS2005, Edmonton, Canada (2005) pp. 3433–3440.

8. J. C. Trinkle and R. J. Milgram, "Complete path planning for closed kinematic chains with spherical joints," *Int. J. Robotics Res.* **21**(9), 773–789 (2002).
9. J.-P. Merlet, "A generic trajectory verifier for the motion planning of parallel robots," *J. Mech. Des.* **123**(4), 510–515 (2001).
10. J.-P. Merlet, "Guaranteed in-the-workspace improved trajectory/surface/volume verification for parallel robots," **In** *Proceedings of the 2004 IEEE International Conference on Robotics and Automation*, New Orleans, LA (2004) pp. 4103–4108.
11. C. K. K. Jui and Q. Sun, "Path trackability and verification for parallel manipulators," **In** *Proceedings of the 2003 IEEE International Conference on Robotics and Automation*, Taipei, Taiwan (2003) pp. 4336–4341.
12. B. Dasgupta and T. S. Mruthyunjaya, "Singularity-free path planning for the stewart platform manipulator," *Mech. Machine Theory* **33**(6), 711–725 (1998).
13. A. K. Dash, I.-M. Chen, S. H. Yeo and G. Yang, "Workspace generation and planning singularity-free path for parallel manipulators," *Mech. Machine Theory* **40**(7), 776–805 (2005).
14. B. Denkena, B. Heimann, H. Abdellatif and C. Holz, "Design, modeling and advanced control of the innovative parallel manipulator palida," **In** *Proceedings of the 2005 IEEE/ASME International Conference on Advanced Intelligent Mechatronics*, AIM2005, Monterrey, USA (2005) pp. 632–637.
15. M. de Berg, M. de Kreveld and M. Overmars, *Computational Geometry, Algorithms and Applications*, 2nd ed. (Springer-Verlag, Berlin, 2000).
16. J.-P. Merlet, *Parallel Robots, Solid Mechanics and Its Applications* (Kluwer Academic Publishers, Dordrecht, 2000).
17. K. Kondo, "Motion planning with six degrees of freedom by multistrategic bidirectional heuristic free-space enumeration," *IEEE Trans. Robotics Automation* **7**(3), 267–277 (1991).
18. P. Isto, "A two-level search algorithm for motion planning," **In** *Proceedings of the 1997 IEEE International Conference on Robotics and Automation*, Albuquerque, USA (1997) pp. 2025–2031.
19. A. K. Dash, I.-M. Chen, S. H. Yeo and G. Yang, "Singularity-free path planning of parallel manipulators using clustering algorithm and line geometry," **In** *Proceedings of the 2003 IEEE International Conference on Robotics and Automation*, Taipei, Taiwan (2003) pp. 761–766.
20. J.-P. Merlet, "Trajectory verification of parallel manipulators in the workspace," **In** *Proceedings of the 1994 IEEE International Conference on Robotics and Automation*, New Orleans, LA (1994) pp. 2166–2171.
21. I. Bonev, Geometric analysis of parallel mechanisms, Ph.D. thesis, Faculté des études supérieures de l'Université Laval (Nov. 2002).
22. C. Gosselin and J. Angeles, "Singularity analysis of closed-loop kinematic chains," *IEEE Trans. Robotics Automation* **6**(3), 281–290 (1990).
23. H. Abdellatif and B. Heimann, "Adapted time-optimal trajectory planning for parallel manipulators with full dynamics modelling," **In** *Proceedings of the 2005 IEEE International Conference on Robotics and Automation*, Barcelona, Spain (2005) pp. 413–418.