Motion Planning for a Novel Reconfigurable Parallel Manipulator with Lockable Revolute Joints

Patrick Grosch, Raffaele Di Gregorio, Javier López, and Federico Thomas

Abstract—This paper introduces a class of reconfigurable parallel robots consisting of a fixed base and a moving platform connected by serial chains having RRP$_S$ (Revolute-Revolute-Prismatic-Spherical) topology. Only the prismatic joint is actuated and the first revolute joint in the chain can be locked or released online. The introduction of these lockable joints allow the prismatic actuators to maneuver to approximate 6-DoF motions for the moving platform. An algorithm for generating these maneuvers is first described. Then, a motion planner, based on the generation of a Probabilistic RoadMap (PRM) whose nodes are connected using the described maneuvers, is presented. The generated trajectories avoid singularities and possible collisions between legs. (See accompanying video)

I. INTRODUCTION

Over the past half-century, the Gough-Stewart platform has been applied extensively to automate many different tasks due to its well-known merits in terms of speed, rigidity, dynamic bandwidth, accuracy, cost, etc. [1]. Since the Gough-Stewart platform has 6 DoF (degrees of freedom), some limited-DoF parallel robots have been designed for applications which do not require full mobility with the aim of simplifying the structure and the control of the general Gough-Stewart platform but without losing its aforementioned merits.

The Gough-Stewart platform consists of a base and a moving platform connected by six UPS (Universal-Prismatic-Spherical) legs, where the underline indicates that the prismatic joint is actuated. Thus, it is usually referenced as a 6-UPS platform. If a number of these UPS legs is eliminated, the mobility of as many of the remaining legs must be reduced by one DoF each, at the same time, to keep the platform location controllable. The resulting parallel manipulator will have a number of DoF equal to the number of the remaining legs. The substitution of the UPS legs with R$_b$RPS legs, where R$_b$ denotes a revolute joint lockable at any time during operation through a brake, is one possibility for implementing this mobility reduction. In fact, each R$_b$RPS will behave as a RPS chain when the R$_b$ joint is locked and, by properly arranging the axis of the revolute joints, as a UPS when it is not. The maximum number of leg eliminations is three, and as many are the manipulator families that can be generated from the Gough-Stewart platform with this technique. Table I summarizes the situation. The 5R$_b$RPS and 4R$_b$RPS architectures are of interest because their motion possibilities can be increased by on-line switching the locked joints. Two kinds of reconfigurable parallel platforms with low mechanical complexity are thus obtained. The architecture involving four legs is probably the most attractive because it uses less actuators (see Fig. 1). This paper is devoted to its study.

The use of lockable joints is not new in Robotics. They

Fig. 1. The proposed platform consists of four R$_b$RPS legs attached to the base through passive lockable revolute joints (top). Since two brakes must be locked at any time to keep the platform rigidly linked to the base, it behaves as a reconfigurable 2RPS-2UPS platform (bottom).
TABLE I

<table>
<thead>
<tr>
<th># of legs (DoF)</th>
<th># of locked joints</th>
<th>Architecture</th>
<th>Related references</th>
</tr>
</thead>
<tbody>
<tr>
<td>6</td>
<td>0</td>
<td>6UPS</td>
<td>[2]</td>
</tr>
<tr>
<td>5</td>
<td>1</td>
<td>4UPS + RPS (reconfigurable)</td>
<td>[3]</td>
</tr>
<tr>
<td>4</td>
<td>2</td>
<td>2UPS + 2RPS (reconfigurable)</td>
<td>[4], [5], [6]</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>3RPS</td>
<td>[7], [8]</td>
</tr>
</tbody>
</table>

have been used at least in [9], [10], and [11].

This paper is organized as follows. Section II studies the kinematics of the proposed platform. Section III shows how to maneuver to locate the platform in any arbitrary pose. Section IV shows how to generate a roadmap in the configuration space of the platform that permits to obtain paths, far from singularities and leg collisions, connecting two arbitrary poses. Section V describes practical aspects concerning the implemented prototype. Finally, the main results are summarized in Section VI.

II. KINEMATICS OF THE 2RPS-2UPS PARALLEL ROBOT

If the leg lengths of the robot in Fig. 1 are fixed, points \( P_i \) and \( P_2 \) are allowed to move on circular arcs, while \( P_3 \) and \( P_4 \) are constrained to move on spheres. The resulting 2RS-2US parallel structure is shown in Fig. 2. With reference to this figure, points \( P_i \) for \( i = 1, \ldots, 4 \) are the centers of the spherical pairs. Points \( A_i \) for \( i = 1, \ldots, 4 \) are the projections of the corresponding \( P_i \) point onto the revolute-pair axes adjacent to the spherical pair, whose \( P_i \) is the center. \( A_3 \) and \( A_4 \) are also chosen as centers of the two universal joints without losing generality. Points \( B_i \) and \( B_3 \) are the projections of \( P_3 \) and \( P_4 \), respectively, onto the line through \( P_1 \) and \( P_2 \). The \( i \)-th leg length \( |P_i - A_i| \) will be denoted \( d_i \), the magnitude of the vector \((P_2 - P_1)\) will be denoted \( a \), whereas the magnitudes of the vectors \((P_j - B_j)\) for \( j = 3, 4 \) will be denoted \( r_j \). Moreover, the following unit vectors and scalar are defined

\[
\mathbf{h}_i = \frac{(P_i - A_i)}{d_i}, \quad i = 1, \ldots, 4;
\]

\[
\mathbf{q} = \frac{(P_1 - A_2)}{|P_1 - A_2|};
\]

\[
\mathbf{w}_1 = \mathbf{u}_1 \times \mathbf{v}_1;
\]

\[
\mathbf{w}_2 = \mathbf{u}_2 \times \mathbf{v}_2;
\]

\[
\mathbf{u}_3 = \frac{(P_2 - P_1)}{a};
\]

\[
\mathbf{v}_3 = \frac{\mathbf{q} \times \mathbf{u}_3}{|\mathbf{q} \times \mathbf{u}_3|};
\]

\[
\mathbf{w}_3 = \mathbf{u}_3 \times \mathbf{v}_3;
\]

\[
b_i = (B_3 - P_1) \cdot \mathbf{u}_3
\]

\[
b_4 = (B_4 - P_1) \cdot \mathbf{u}_3
\]

A. Position Analysis

The determination of the actuated-joint variables (leg lengths) for an assigned pose of the platform [Inverse Position Analysis (IPA)] is straightforward. In fact, once the positions of \( P_i \), for \( i = 1, \ldots, 4 \), are known, the leg lengths can be immediately computed since the positions of \( A_i \), for \( i = 1, \ldots, 4 \), are geometric data linked to the base reference frame [see Fig. 1(bottom)]. On the contrary, the determination of the platform pose for assigned leg lengths [Forward Position Analysis (FPA)] requires the solution of the 2RS-2US’ closure equations which constitute a nonlinear equation system. This problem coincides with the one encountered when solving the FPA of the 6-4 fully-parallel mechanism [4] since that mechanism generates an 2RS-2US structure when the actuated joints are locked [see Fig. 1(bottom)]. In [4], Innocenti gave the analytical solution of this problem and showed that, in general, up to 32 platform poses may be compatible with an assigned set of leg lengths. In the following part of this subsection, the 2RS-2US’ closure equations will be deduced in a form slightly different, from the one reported in [4], which is more appropriate to the analysis presented in the next subsection. With reference to Fig. 2 and the adopted notations, the 2RS-2US’ closure equations can be written as follows:

\[
(P_2 - P_1) \cdot (P_2 - P_1) = a^2 \quad (1)
\]

\[
(P_3 - A_3) \cdot (P_3 - A_3) = d_3^2 \quad (2)
\]

\[
(P_4 - A_4) \cdot (P_4 - A_4) = d_4^2 \quad (3)
\]
where
\[
P_1 = A_1 + d_1(c_1v_1 + s_1w_1) \quad (4)
\]
\[
P_2 = A_2 + d_2(c_2v_2 + s_2w_2) \quad (5)
\]
\[
P_3 = P_1 + b_3u_3 + r_3(c_3v_3 + s_3w_3) \quad (6)
\]
\[
P_4 = P_1 + b_4u_3 + r_4(c_4v_3 + s_4w_3) \quad (7)
\]

Equation (1) is a trigonometric c-s-linear equation that involves only \(c_i, s_i, v_i, w_i\) for \(i = 1, \ldots, 4\) stand for \(\cos(\theta_i)\) and \(\sin(\theta_i)\), respectively.

B. Singularities

The configurations where the platform can perform elementary motions, even though the actuators are locked, are called parallel singularities. Parallel singularities are critical both from the control (the platform pose becomes no longer controllable) and the statics (some links should stand infinite internal loads) point of views. Thus, they must be avoided during operation. When the 2RPS-2UPS platform is at a parallel singularity, the 2RS-2US structure obtained by locking the actuators is singular, too (i.e., the structure is not rigid). Thus, by looking for the 2RS-2US singular geometries, the parallel singularities of the associated 2RPS-2UPS can be found.

When the 2RS-2US structure assumes a singular geometry, the platform can perform elementary motions that must fulfill the following velocity equations, deduced by differentiating equations (1), (2), and (3):
\[
\hat{P}_2 - \hat{P}_1 \cdot u_3 = 0 \quad (10)
\]
\[
\dot{P}_3 \cdot h_3 = 0 \quad (11)
\]
\[
\hat{P}_4 \cdot h_4 = 0 \quad (12)
\]

where
\[
\hat{P}_1 = \hat{\theta}_1 d_1(u_1 \times h_1)
\]
\[
\hat{P}_2 = \hat{\theta}_2 d_2(u_2 \times h_2)
\]
\[
\hat{P}_3 = \hat{\theta}_3 d_1(u_1 \times h_1) + \frac{b_3}{a} [\hat{\theta}_2 d_2(u_2 \times h_2) - \hat{\theta}_1 d_1(u_1 \times h_1)]
\]
\[
+ \hat{\theta}_3 [u_3 \times (P_3 - B_3)] + r_3 [c_3v_3 + s_3w_3]
\]
\[
\hat{P}_4 = \hat{\theta}_1 d_1(u_1 \times h_1)
\]
\[
+ \frac{b_4}{a} [\hat{\theta}_2 d_2(u_2 \times h_2) - \hat{\theta}_1 d_1(u_1 \times h_1)]
\]
\[
+ \hat{\theta}_3 [u_3 \times (P_4 - B_3)] + r_4 [c_4v_3 + s_4w_3]
\]

Equation (1) is a trigonometric c-s-linear equation that involves only \(c_i, s_i, v_i, w_i\) for \(i = 1, \ldots, 4\) stand for \(\cos(\theta_i)\) and \(\sin(\theta_i)\), respectively. Parallel singularities are singularities of the internal RSSR loop and, in general, they are singularities of the 2RS-2US structure, too. The introduction of (13) and (14) into (10) yields
\[
\begin{align*}
\begin{bmatrix}
\hat{m}_{11} & \hat{m}_{12} \\
\hat{m}_{21} & \hat{m}_{22}
\end{bmatrix}
\begin{bmatrix}
\hat{\theta}_1 \\
\hat{\theta}_3
\end{bmatrix}
= 0
\end{align*}
\]

System (22) admits a non-null solution for \(\hat{\theta}_1 \) and \(\hat{\theta}_3\) (i.e., a singular configuration occurs for the 2RS-2US structure) if and only if
\[
m_{11}m_{22} - m_{12}m_{21} = 0.
\]

The above relationship is the analytic expression of the singularity condition of the 2RS-2US structure. It is satisfied either when the two bi-dimensional column vectors \(m_i = (m_{1i}, m_{2i})^T\), for \(i = 1, 2\), are parallel or when at least one of the \(m_i\) vectors is a null vector. The dimensionless parameters
\[
k_1 = \frac{|m_1|}{|m_2|}, \quad k_2 = \frac{|m_1||m_2|}{|m_1 \cdot m_2|}
\]

can be used to evaluate how far from singularity a configuration is. The farthest-from-singularity configuration is the one where \(k_1\) is equal to 1 and \(k_2\) is equal to infinity; whereas a singular configuration occurs when at least one of the following conditions occur: (a) \(k_1\) is equal to 0, (b) \(k_1\) is equal to infinity, (c) \(k_2\) is equal to 1. Based on these values, the following objective function, to be maximized during platform motion, can be defined
\[
n = \frac{k_1}{(k_1 - 1)^2} + (k_2 - 1).
\]

Such a function goes toward infinity when \(k_1 \) \((k_2)\) goes toward 1 (infinity); and it decreases when either \(k_1 \) \((k_2)\) goes
toward zero 0 (1) or $k_1$ goes toward infinity. It will be useful later, when assigning a cost to a path.

III. MANEUVERS

Let us assume that we want to generate a trajectory connecting $X_0 = (L_0, \Phi_0) = (d_{01}^0, \ldots, d_{04}^0, \phi_{01}^0, \ldots, \phi_{04}^0)$ to $X_f = (L_f, \Phi_f) = (d_{11}^f, \ldots, d_{14}^f, \phi_{11}^f, \ldots, \phi_{14}^f)$ where $d_i$ is the length of leg $i$ and $\phi_1$ is the angle formed by $g_i$ and the $x$–axis of the world reference frame (see Fig. 1). Since the robot is not capable, in general, of reaching the final pose directly, it is necessary to introduce an intermediate one (a via pose) where the lockable joints are switched. The leg lengths in the via pose, say $L_x$, can be computed numerically by setting the released joints to their values in the final pose and solving a local optimization problem starting from the initial pose. This can be efficiently implemented using the Newton’s method [12]. Then, the proposed maneuver consists in the four steps detailed in Fig. 3.

Note that there are up to six sets of possible maneuvers connecting two given poses: one for each possible pair of locked joints. Once we have a candidate for a maneuver, and its corresponding via pose, it must be executed by driving the robot’s prismatic actuators, as explained above. The simplest driving law is that consisting in linearly interpolating the leg lengths from $L_0$ to $L_x$, and then from $L_x$ to $L_f$. During this process, it might happen that the system reaches a different solution from the expected one (remind that the forward kinematics problem has no single solution). If so, the generated maneuver is not valid. This might happen mainly when the maneuver involves a path close to a singularity. For the sake of simplicity, in this case the obtained maneuver would not be considered as valid, though a more sophisticated driving law might connect the initial to the final configuration through the obtained via pose.

There is one more reason to reject a candidate for a maneuver: it leads to collisions or the joints are not kept within their valid range of motion. A complete test for collision detection can be implemented using available collision detection packages such as GJK, SOLID, V-Clip, I-Collide, etc. (see [15, p. 201] and the references therein).

Once all valid maneuvers are computed, it is reasonable to choose the one that keeps the platform as far as possible of its parallel singularities. Unfortunately, there is no proper distance to a singularity [13]. As a simplification in our particular design, the quality measure to decide whether the maneuver is close to a singularity is taken to be (25) evaluated in the corresponding via pose. It is assumed that the bigger this value is, the farther the via pose is from a singularity. Then, the reciprocal of this value is taken as the cost of a maneuver.

The above procedure to find the best maneuver connecting two arbitrary configurations is summarized in pseudocode in Algorithm 1. Function Candidate implements the Newton’s method that computes the leg lengths in the via pose. Function ValidPath verifies if the final configuration is reached by linearly interpolating the leg lengths, checks if no collisions arise, and verifies if the joints are kept within their range along the trajectory. Finally, function Cost assigns a cost to the maneuver based on the objective function (25) to a singularity of the via pose.

It is clear that the above algorithm might fail to find a path mainly when the initial and final poses are far apart in the configuration space of the robot. In these cases, one alternative is to subdivide the trajectory into segments whose initial and final poses can be connected using the above algorithm. Unfortunately, this simple idea might also fail. The alternative is to use a motion planner, as described in the next section.
Algorithm 1 BestViaPose(Xi, Xj)
1: Maneuvers ← \{[1,2,3,4],[1,3,2,4],[1,4,2,3],[2,3,1,4],
2: ...
3: /* The first two indices of each 4-tupla correspond to the
locked joints during the first motion of the maneuver */
4: Xv ← void
5: for all M ∈ Maneuvers do
6:   [i, j, k, l] ← M
7:   \(\Phi_x[i] ← \Phi_0[i]\)
8:   \(\Phi_x[j] ← \Phi_0[j]\)
9:   \(\Phi_x[k] ← \Phi_0[k]\)
10:  \(\Phi_x[l] ← \Phi_f[l]\)
11:  \(L_x ← \text{Candidate}(X_0, \Phi_x, M)\)
12:  \(X_x ← (L_x, \Phi_x)\)
13:  if ValidPath(X0, Xx, Xf) = TRUE then
14:     /* The maneuver is valid */
15:        if Cost(Xx) < Cost(Xv) then
16:            /* Cost(void) returns \(\infty\) */
17:              \(X_v ← X_x\)
18:        end if
19:    end if
20: end for
21: return \(X_v\)

IV. PATH PLANNING

There are many possible approaches for implementing a motion planner but those based on Probabilistic RoadMaps (PRM) [14] have demonstrated their tremendous potential in many applications [15, Chapter 7]. This approach has already been successfully applied to ordinary parallel robots in [16]. Next, it is adapted to the proposed reconfigurable robot. Within this approach, the proposed robot would be subjected to a learning phase where its configuration space is randomly sampled. These samples are connected to their neighbors through the maneuvers, presented in the previous section, to generate a roadmap. Then, in the query phase, in which a path between two arbitrary poses must be found, the initial and final poses are firstly linked to their neighbors in the roadmap and, using a graph search algorithm, the best path according to a given criterion is found.

A. Generating the roadmap

The roadmap is built by sampling poses in the configuration space of the robot. When a sample is chosen, the best maneuvers to connect it to its neighboring poses previously generated are computed. Two poses are considered to be neighbors if the Euclidian norm between their position and orientation components are below a given threshold. If a valid maneuver is found, its corresponding via pose is stored in an adjacent matrix together with its associated cost. If not, the stored cost will be infinite. Algorithm 2 gives this description in pseudocode.

To increase the density of the roadmap, it is always possible to add an intermediate configuration when two configurations fail to be connected directly through one of the six maneuvers that can be obtained using the procedure described above.

B. Finding a path

If a trajectory —free from collisions and as far as possible from any singularity— connecting \(X_0\) to \(X_f\) must be generated, it is firstly necessary to connect these two poses to the previously generated roadmap. That is, the best maneuvers to connect them to their neighbors should be computed. Once the initial and final poses are connected to the roadmap, it is only needed to find the shortest path connecting them in terms of costs. Dijkstra’s algorithm is well-suited to this end [17, p. 595]. Finally, when the path is obtained, if one exists, the corresponding maneuvers —described in terms of leg lengths settings and sequences of locked and released revolute joints— can be executed by the robot.

V. IMPLEMENTATION

In order to verify the behavior of the proposed parallel robot and the described path planner, a simulator using MATLAB and Simulink whose output is connected to a VMRL 3D model of the robot was implemented. Using the equations presented in Section II, it simulates the motion of the platform generated by applying the leg lengths settings and the sequence of switchings obtained by the path planner. A typical output of this simulator can be seen in the attached video.

The diameters of the base and the platform are 0.4m and 0.2m, respectively. When the legs are extended at half their maximum extension, the platform is located at 0.3m from the base. This is taken as the home configuration. The generated roadmap has been obtained by taking 100 configurations randomly sampled in a region centered at this configuration with \(x \in [-0.04, 0.04], y \in [-0.04, 0.04], z \in [0.115, 0.125], \theta_x \in [-0.05, 0.05], \theta_y \in [-0.05, 0.05], \text{and } \theta_z \in [-0.05, 0.05] \) where distances are given in meters and the orientation angles in radians using the roll-pitch-yaw convention). When each of these configurations have been tried to be connected to all others, 2,275 connections fail (out of the 4,950 possible connections) for the six possible maneuvers. If an intermediate configuration is added in these cases, the amount of failed connections drops to 644. Due to these intermediate configurations, the total number of configurations in the roadmap is 3229 and the total number of
maneuvers checked for validity amounts to 114,420. 92,245 are discarded for different reasons. Table II compiles this information.

<table>
<thead>
<tr>
<th>Connection Type</th>
<th>Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>Evaluated connections</td>
<td>19070</td>
</tr>
<tr>
<td>Possible direct connections</td>
<td>4950</td>
</tr>
<tr>
<td>Failed direct connections</td>
<td>2275</td>
</tr>
<tr>
<td>Failed after adding one intermediate configuration</td>
<td>644</td>
</tr>
<tr>
<td>Established connections</td>
<td>9356</td>
</tr>
</tbody>
</table>

After verifying the behavior of the proposed robot in simulation, the prototype in Fig. 4 was built. The base and the moving platform are made of 3mm thick nickel-plated steel plates. They are disks of 400 mm and 200 mm in diameter, respectively. The lockable revolute joints have been implemented using electromagnetic brakes. When one of this brakes is energized, the corresponding axis of rotation is released, otherwise it remains locked. The actuated prismatic joints are implemented using miniature servo linear motors. The four actuators are controlled through a USB servo card. An interesting feature of this prototype is that the legs are attached to the base and the moving platform through magnetic fixtures. This simplifies any leg rearrangement during tests. Finally, all plastic elements were manufactured using a 3D printer.

VI. CONCLUSIONS

By introducing legs of type R\textsubscript{b}R\textsubscript{PS}, where R\textsubscript{b} stand for a lockable revolute joint, two novel reconfigurable parallel robots of reduced mechanical complexity —the 5R\textsubscript{b}R\textsubscript{PS} and the 4R\textsubscript{b}R\textsubscript{PS}— have been proposed. Moreover, the 4R\textsubscript{b}R\textsubscript{PS} has been studied in depth, and a practical implementation of it has been presented.

Regarding the 4R\textsubscript{b}R\textsubscript{PS}, it has been demonstrated that: (i) its moving platform can be moved in a six-dimensional operational space by using only four actuators that are maneuvered so that via poses, where the couple of locked R\textsubscript{b} pairs is changed, are introduced; (ii) the parallel singularities can be avoided and the maximum forces in the actuators can be reduced by suitably managing the insertion of via poses. Eventually, these theoretical results have been verified on the built prototype.

REFERENCES