sensors

MDPI

Article

Depth-Dependent Control in Vision-Sensor Space for Reconfigurable Parallel Manipulators

Arturo Franco-López [1], Mauro Maya [1,*], Alejandro González [2], Antonio Cardenas [1] and Davide Piovesan [3]

[1] Facultad de Ingenieria, Universidad Autonoma de San Luis Potosi, San Luis Potosi 78290, Mexico; arturo.franco@uaslp.mx (A.F.L.); antonio.cardenas@uaslp.mx (A.C.)
[2] Escuela de Ingeniería y Ciencias, Tecnologico de Monterrey, Querétaro 76130, Mexico; a.gonzalezda@tec.mx
[3] Biomedical, Industrial and Systems Engineering Department, Gannon University, Erie, PA 16541, USA; piovesan001@gannon.edu
* Correspondence: mauro.maya@uaslp.mx

Abstract: In this paper, a control approach for reconfigurable parallel robots is designed. Based on it, controls in the vision-sensor, 3D and joint spaces are designed and implemented in target tracking tasks in a novel reconfigurable delta-type parallel robot. No a priori information about the target trajectory is required. Robot reconfiguration can be used to overcome some of the limitations of parallel robots like small relative workspace or multiple singularities, at the cost of increasing the complexity of the manipulator, making its control design even more challenging. No general control methodology exists for reconfigurable parallel robots. Tracking objects with unknown trajectories is a challenging task required in many applications. Sensor-based robot control has been actively used for this type of task. However, it cannot be straightforwardly extended to reconfigurable parallel manipulators. The developed vision-sensor space control is inspired by, and can be seen as an extension of, the Velocity Linear Camera Model–Camera Space Manipulation (VLCM-CSM) methodology. Several experiments were carried out on a reconfigurable delta-type parallel robot. An average positioning error of 0.6 mm was obtained for static objectives. Tracking errors of 2.5 mm, 3.9 mm and 11.5 mm were obtained for targets moving along a linear trajectory at speeds of 6.5, 9.3 and 12.7 cm/s, respectively. The control cycle time was 16 ms. These results validate the proposed approach and improve upon previous works for non-reconfigurable robots.

Keywords: vision-based control; camera-space manipulation; parallel robot

Citation: Franco López, A.; Maya, M.; González, A.; Cardenas, A.; Piovesan, D. Depth-Dependent Control in Vision-Sensor Space for Reconfigurable Parallel Manipulators. *Sensors* **2022**, *23*, 7039. https://doi.org/10.3390/s23167039

Academic Editors: Shuai Li, Dechao Chen, Vasilios N. Katsikis, Predrag S. Stanimirovic, Dunhui Xiao and Mohammed Aquil Mirza

Received: 21 June 2023
Revised: 25 July 2023
Accepted: 2 August 2023
Published: 9 August 2023

1. Introduction

Parallel robots have received increased attention in recent years, and one of the most popular applications at an industrial level is pick and place based on the parallel delta robot [1]. A parallel robot can be described as a manipulator composed of a fixed and a mobile platform connected by two or more kinematic chains generating closed loops with the end effector mounted on the mobile platform [2,3]. The payload is therefore distributed among the kinematic chains; besides, the actuators are mounted directly in the fixed base, which means that the kinematic chains do not carry them. This results in thinner and lighter links. Parallel robots have some advantages over serial robots: higher positioning accuracy, better rigidity, greater load-capacity-to-weight ratio, higher speeds and accelerations, lower inertia, good stability, and simple inverse kinematics [3–6]. On the other hand, parallel robots have some disadvantages: relatively small and complex workspace, often having complex singularities inside the workspace: complex geometric (position relationships), kinematic (velocity relationships) and dynamic models that complicate their modeling and control design [5,7,8].

1.1. Reconfiguration and Redundancy on Parallel Robots

Reconfiguration has the ability to facilitate the adaptation and optimization of parallel robots to a wide range of applications. It overcomes some of their typical difficulties such as the reduced workspace. Reconfiguration on parallel robots may involve changes to the length of the arms, the orientation of the kinematic chains, or the position of the mobile platform in relation to the fixed base. These changes can be made manually or automatically using intelligent control systems. The implementation of a reconfigurable robot in which the reconfiguration is performed via a mechanism that can be controlled in operation time makes the robot redundant [9,10], which requires an even more complex control design.

In parallel robots, there are multiple types of redundancy, the most important being the kinematic redundancy and the actuator redundancy. A way to generate kinematic redundancy consists of adding active joints to the kinematic chains [11–14]. This implies that there is an infinite combination of joint velocities producing the same specific velocity of the end effector. Actuator redundancy consists of having more than the necessary kinematic chains between the fixed and mobile platforms. From the static point of view, for each force vector and torque at the end effector, there will be an infinite number of force and wrench vectors for the active joints [15]. For redundant robots, obtaining a single solution to the inverse kinematic problem is called solving the redundancy problem.

Redundancy has great advantages in parallel robots, such as enhancing the workspace, avoidance or reduction of singularities, increasing the payload capacity, improving the dynamics of the mechanism, and eliminating motion clearance, among others [5,16,17]. However, these advantages come at a price such as increased complexity in the kinematic and dynamic models, challenging control design, and higher energy consumption due to extra actuators and internal forces/torques [18]. Moreover, extra actuators imply extra costs and introduce additional friction from the supplementary active joints. In summary, robot reconfiguration can be used to tackle some of the problems of parallel robots at the cost of increasing the complexity of the manipulator, resulting in a more challenging control design.

1.2. Camera Space Manipulation

Providing parallel robots with sensors that enable them to monitor the environment around them is another effective method of extending the range of applications for these machines. With this information, the robot is able to adapt to its environment and the events occurring in it. In particular, for an object-tracking task, it is important for a robot to be able to respond appropriately to changes in the environment. Vision systems are one of the most frequently used options to obtain information about the interaction of the robot with the environment due to their ease of operation [19]. Vision-based control is a sensor-based control strategy, strongly resembling the process of our central nervous system [20]. Among the control schemes that use vision systems, two of the most general and popular are Visual Servoing (VS) [21–23] and Camera Space Manipulation (CSM) [24,25]. VS control is commonly applied for real-time control, incorporating visual information from one or more cameras in a closed-loop control cycle. It provides increased precision in manipulation tasks based on visual feedback [22,26,27]; it also requires the Image Jacobian, which depends on the visual features used and the camera parameters (obtained through a calibration process) and can be complex and difficult to obtain [23]. In its first applications, the CSM methodology [24] and its variants, such as Linear Camera Model–Camera Space Manipulation (LCM-CSM) [28] consisted of an open-loop position level control. One of the most interesting features is that it is not necessary to know the view parameters of the camera a priori. Instead, a data capture is performed in which either a set of points in three-dimensional coordinates or a set of joint coordinates are mapped to their corresponding coordinates in image space to obtain such parameters. A practical way to accomplish this is by placing visual markers on the robot's end effector and using the geometric model of the camera to associate the coordinates of the markers with their corresponding coordinates in camera space. This circumvents the need to know a priori the Jacobian transformation of

the manipulator, which can be a complex task [29,30]. With the data set and the camera model, the estimation of the so-called vision parameters is performed. This process is called "preplan". An even simpler method is to approximate the orthographic camera model with the pin-hole camera model obtaining the LCM-CSM [28] variant where the view parameters are estimated from a linear model. Recent work has derived a camera–Jacobian matrix from calculating the derivative with respect to time for a pin-hole camera model. This allowed the development of the VLCM-CSM variant enabling the design of velocity-based closed-loop control laws [6].

Some vision techniques have been reported in the literature to guide parallel robots through various tasks. In [29], a vision system is used to estimate the pose of the Hexa parallel robot, while [31] uses a computer vision system to develop an obstacle collision detection method for the same robot, no control description is included in either work. In [32], computer vision is used to identify the geometric parameters of a Delta parallel architecture. Regular PIDs are then used to control the robot joints. Ref. [33] uses computer vision and an unscented Kalman filter to estimate the pose of a planar 3RRR parallel robot and uses a PID control to guide the robot through a point to point path-tracking task. In [34,35], VS is used to control a cable-driven parallel robot, pose-based (3D and 2D) schemes are developed as well as a mix of them (2D 1/2); a conservative sufficient condition for stability is obtained. Ref. [6] develops the VLCM-CSM approach to control a Delta-type parallel robot in target tracking tasks. In summary, there are still few works developing vision-based control for parallel robots, some of them are used only to detect the end-effector pose to then guide it to complete the task. However, to the best of the authors' knowledge, no general control design approach for reconfigurable parallel robots has been reported in the literature.

This work presents a control approach for reconfigurable parallel robots. Following this approach, controls are designed and implemented in a novel reconfigurable Delta-type parallel robot [16]. In this work. the geometric and kinematic models are derived for this robot for the first time, yielding an initial contribution of this paper. Moreover, the control design in image space is based on the VLCM-CSM approach. For this, an original derivation of the image Jacobian associated with this approach is obtained, where it is shown that the only variable is a factor depending on the depth of the observed feature. This makes this Jacobian extremely simple to implement and update on each control cycle. This simple Jacobian can be used for any variant of the VLCM-CSM methodology and constitutes a second contribution of this work. Using the aforementioned elements, a general control approach, and main contribution of this work, for reconfigurable parallel robots was developed. Based on this approach, control laws in the the vision-sensor, 3D and joint spaces are designed and implemented. The controls in the 3D and joint spaces have the limitation that it is in general difficult to obtain the reference information (target position) in such spaces, particularly in the case of a target moving along an unknown trajectory. Usually this information has to be obtained through indirect estimates using available measurements and position reconstruction models prone to errors. This limitation is intrinsic to any control design in these spaces. The control in the vision sensor space does not require reconstruction as this type of sensor provides direct measurements of the target in that space regardless of its trajectory, this results in a robust approach.

This article is organized as follows: Section 2 analyzes the kinematic model of the reconfigurable delta robot considered in this work. Section 3 presents an original derivation of the image Jacobian based on the VLCM-CSM methodology. In Section 4, a control approach for reconfigurable parallel robots is developed and several control laws based on this approach are synthesized for a reconfigurable delta-type parallel robot. Section 5 details the experimental setup and the experiments performed. Section 6 presents and discusses the results. Finally, Section 7 describes the authors' conclusions of this work and outlines opportunities for future work.

2. Modeling of the Reconfigurable Delta Robot

2.1. Description of the Reconfigurable Delta Robot

Let us consider the delta robot in Figure 1; it consists of two platforms, one fixed and a mobile one, connected by three kinematic chains (numbered with index $i = 1, 2, 3$), evenly distributed about the vertical axis Z of the robot's fixed frame. In this typical configuration, the fixed platform is on top of the mechanism (horizontal element), and the mobile platform (i.e., the part where the end-effector is attached) is a small, horizontal element at the bottom of the mechanism. Each chain has two links; one end of the chain is connected to the fixed platform through an actuated, rotational joint placed at point A_i with joint variable θ_{i1}, and the other end is connected to the mobile platform through a passive universal joint (some designs use a spherical one), at point C_i. The two links on each chain are connected together at point B_i through a passive universal (can be spherical) joint with (passive) joint variables θ_{i2} and θ_{i3}. The use of links with a parallelogram configuration, together with the robot architecture, ensures the parallelism between both platforms at all times. R is the radius of the fixed platform (i.e., the distance between the origin of the robot's frame and point A_i), r is the radius of the moving platform, (i.e., the distance between the center of the platform and point C_i), α_i is the angle between axis X of the fixed frame and each of the three kinematic chains that make up the robot, measured about the fixed Z axis; L_1 and L_2 are the length of the first and second links of the kinematic chains, respectively, and P is the 3D end-effector position expressed in the fixed frame. With the reconfiguration capability considered in this work, R becomes a (joint) variable. In Figure 2, the reconfiguration mechanism for the robot is shown. More details on the reconfiguration analysis and mechanism can be found in [16].

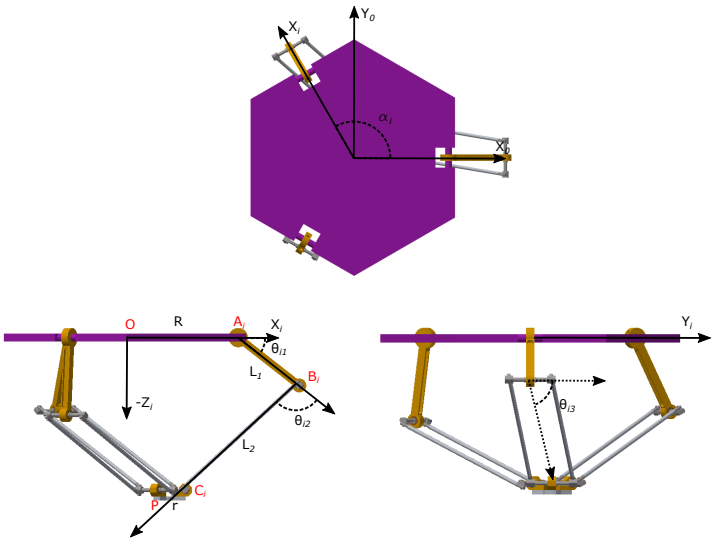

Figure 1. Kinematic chains of the delta robot.

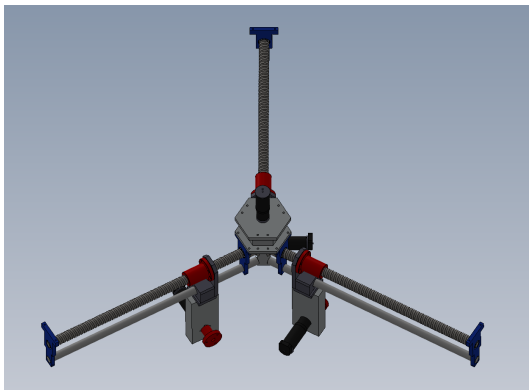

Figure 2. Reconfiguration mechanism for the delta robot.

The following key points can be defined with respect to the robot's fixed reference frame:

$$\mathbf{A}_i = \begin{bmatrix} R\cos(\alpha_i) \\ R\sin(\alpha_i) \\ 0 \end{bmatrix} \tag{1}$$

$$\mathbf{B}_i = \begin{bmatrix} (R + L_1\cos(\theta_{i1}))\cos(\alpha_i) \\ (R + L_1\cos(\theta_{i1}))\sin(\alpha_i) \\ -L_1\sin(\theta_{i1}) \end{bmatrix} \tag{2}$$

$$\mathbf{C}_i = \begin{bmatrix} P_x + r\cos(\alpha_i) \\ P_y + r\sin(\alpha_i) \\ P_z \end{bmatrix} \tag{3}$$

$$\mathbf{P} = \begin{bmatrix} P_x \\ P_y \\ P_z \end{bmatrix} \tag{4}$$

2.2. Forward Geometric Model

Based on the work reported in [16], and considering the reconfiguration, we define the active joints and Cartesian coordinate vectors of the end-effector as

$$\mathbf{q} = \begin{bmatrix} R \\ \theta_{11} \\ \theta_{21} \\ \theta_{31} \end{bmatrix} \tag{5}$$

and

$$\mathbf{x} = \mathbf{P} = \begin{bmatrix} P_x \\ P_y \\ P_z \end{bmatrix} \tag{6}$$

respectively. Note that R is included in the vector of joint variables because the reconfiguration mechanism can be controlled together with the three kinematic chains. On this basis, the reconfigurable robot is then modeled as a redundant robot with four control inputs.

A common way to solve the direct kinematics problem for the delta robot is to solve the intersection of three spheres centered at points $\tilde{\mathbf{B}}_i$ defined as

$$\tilde{\mathbf{B}}_i = \begin{bmatrix} B_{ix} - r\cos(\alpha_i) \\ B_{iy} - r\sin(\alpha_i) \\ B_{iz} \end{bmatrix} \tag{7}$$

and radii L_2. The spheres are associated with the free motion of points C_i with respect to points B_i, and obviously do not intersect at P, however, by horizontally shifting the spheres (their centers) a distance r towards fixed axis Z, the shifted spheres intersect at P. With this, we define a sphere per each kinematic chain with center $\tilde{\mathbf{B}}_i$, and the intersection of the three spheres with the smallest value on the Z axis will be $\mathbf{P} = \mathbf{x}$, which is the solution to the direct kinematics problem. This can be represented by the following system of equations:

$$(\tilde{B}_{ix} - P_x)^2 + (\tilde{B}_{iy} - P_y)^2 + (\tilde{B}_{iz} - P_z)^2 = L_2^2 \tag{8}$$

It may be observed that this approach is valid regardless of whether R is constant or variable. For that, this solution is valid for a reconfigurable delta robot.

2.3. Forward Kinematic Model

Based on Figure 1, the following closed-loop geometric constraint system can be defined:

$$\overrightarrow{\mathbf{OP}} + \overrightarrow{\mathbf{PC_i}} = \overrightarrow{\mathbf{OA_i}} + \overrightarrow{\mathbf{A_iB_i}} + \overrightarrow{\mathbf{B_iC_i}} \tag{9}$$

which has the following matrix form:

$$\begin{bmatrix} P_x\cos(\alpha_i) - P_y\sin(\alpha_i) \\ P_x\sin(\alpha_i) + P_y\cos(\alpha_i) \\ P_z \end{bmatrix} + \begin{bmatrix} r \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} R \\ 0 \\ 0 \end{bmatrix} + L_1 \begin{bmatrix} \cos(\theta_{1i}) \\ 0 \\ -\sin(\theta_{1i}) \end{bmatrix} + L_2 \begin{bmatrix} \sin(\theta_{i3})\cos(\theta_{i2}+\theta_{i1}) \\ \cos(\theta_{i3}) \\ -\sin(\theta_{i3})\cos(\theta_{i3}+\theta_{i1}) \end{bmatrix} \tag{10}$$

This can be rewritten as

$$(\mathbf{x}_i + \mathbf{r}) = \mathbf{R} + \mathbf{L}_{1i} + \mathbf{L}_{2i} \tag{11}$$

Deriving with respect to time, we obtain the following result:

$$\dot{\mathbf{x}}_i = \dot{\mathbf{R}} + \dot{\mathbf{L}}_{\mathbf{1i}} + \dot{\mathbf{L}}_{\mathbf{2i}} \tag{12}$$

for the reconfigurable robot, R has a variable derivative, while in the case of the robot without reconfiguration, the derivative of R is zero.

Notice that the links can only move around a pivot; hence, we can simplify the previous equation in the following form:

$$\mathbf{x}_i = \mathbf{R} + \mathbf{w}_{L_{1i}} \times \mathbf{L_{1i}} + \mathbf{w}_{L_{2i}} \times \mathbf{L_{2i}} \tag{13}$$

To further simplify the equation above it is convenient to express the equation above on a reference frame located on B_i and obtain an expression that depends only on the articular variables. To accomplish this, we multiply the above equation by the director vector $\hat{\mathbf{L}}_{\mathbf{2i}}$, obtaining

$$\hat{\mathbf{L}}_{\mathbf{2i}} \cdot \dot{\mathbf{x}}_i = \hat{\mathbf{L}}_{\mathbf{2i}} \cdot \dot{\mathbf{R}} + \hat{\mathbf{L}}_{\mathbf{2i}} \cdot \left(\mathbf{w}_{L_{1i}} \times \mathbf{L_{1i}}\right) \tag{14}$$

Substituting Equation (10) in the left side of the above equation, we have:

$$\begin{aligned} \hat{\mathbf{L}}_{\mathbf{2i}} \cdot \dot{\mathbf{x}}_i = [\sin\theta_{i3}\cos(\theta_{i2}+\theta_{i1})]\left[\dot{P}_x\cos\alpha_i - \dot{P}_y\sin\alpha_i\right] + \cos\theta_{i3}\left[\dot{P}_x\sin\alpha_i + \dot{P}_y\cos\alpha_i\right] \\ -[\sin\theta_{i3}\sin(\theta_{i2}+\theta_{i1})]\dot{P}_z = a_{ix}\dot{P}_x + a_{iy}\dot{P}_z + a_{iz}\dot{P}_z \end{aligned} \tag{15}$$

where

$$\begin{aligned} a_{ix} &= \sin\theta_{i3}\cos(\theta_{i2}+\theta_{i1})\cos\alpha_i + \cos\theta_{i3}\sin\alpha_i \\ a_{iy} &= -\sin\theta_{i3}\cos(\theta_{i2}+\theta_{i1})\sin\alpha_i + \cos\theta_{i3}\cos\phi_i \\ a_{iz} &= -\sin\theta_{i3}\sin(\theta_{i2}+\theta_{i1}) \end{aligned} \tag{16}$$

Substituting the right side of Equation (14), we have

$$\hat{\mathbf{L}}_{\mathbf{2i}} \cdot \left(\mathbf{w}_{L_{1i}} \times \mathbf{L_{1i}}\right) = -L_1\sin\theta_{i2}\sin\theta_{i3}\dot{\theta}_{i1}$$

and

$$\hat{\mathbf{L}}_{\mathbf{2i}} \cdot \dot{\mathbf{R}} = \dot{R}\sin\theta_{i3}\cos(\theta_{i2}+\theta_{i1})$$

in such a way that

$$\hat{\mathbf{L}}_{\mathbf{2i}} \cdot \dot{\mathbf{R}} + \hat{\mathbf{L}}_{\mathbf{2i}} \cdot \left(\mathbf{w}_{L_{1i}} \times \mathbf{L_{1i}}\right) = \sin\theta_{i3}\cos(\theta_{i2}+\theta_{i1})\dot{R} - L_1\sin\theta_{i2}\sin\theta_{i3}\dot{\theta}_{i1} \tag{17}$$

Substituting in the Equations (16) and (17) for each i we have

$$a_{1x}\dot{P}_x + a_{1y}\dot{P}_y + a_{1z}\dot{P}_z = \sin(\theta_{13})\cos(\theta_{12}+\theta_{11})\dot{R} - L_1\sin(\theta_{12})\sin(\theta_{13})\dot{\theta_{11}}$$

$$a_{2x}\dot{P}_x + a_{2y}\dot{P}_y + a_{2z}\dot{P}_z = \sin(\theta_{23})\cos(\theta_{22}+\theta_{21})\dot{R} - L_1\sin(\theta_{22})\sin(\theta_{23})\dot{\theta_{21}}$$
$$a_{3x}\dot{P}_x + a_{3y}\dot{P}_y + a_{3z}\dot{P}_z = \sin(\theta_{33})\cos(\theta_{32}+\theta_{31})\dot{R} - L_1\sin(\theta_{32})\sin(\theta_{33})\dot{\theta_{31}}$$

which can be rewritten in matrix form as:

$$J_x\dot{\mathbf{x}} = J_q\dot{\mathbf{q}} \tag{18}$$

where

$$J_x = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} \tag{19}$$

is a Jacobian matrix that associate each point B_i with the work-space, and where

$$J_q = \begin{bmatrix} b_{11} & b_{12} & 0 & 0 \\ b_{21} & 0 & b_{23} & 0 \\ b_{31} & 0 & 0 & b_{34} \end{bmatrix} \tag{20}$$

with

$$b_{i1} = \sin(\theta_{i3})\cos(\theta_{i2}+\theta_{i1})$$
$$b_{i,i+1} = -L_1\sin(\theta_{i2})\sin(\theta_{i3})$$

is a Jacobian matrix that associate the joint-space with each B_i.

When the robot is not reconfigured, these Jacobians become square and measure 3×3. The new joint variable R is reflected in the Jacobian J_q and generates a non-square matrix of dimension 4×3 while the matrix J_x is not affected and remain of dimension 3×3.

The following forward kinematic model is obtained from Equation (18):

$$\dot{\mathbf{x}} = \mathbf{J}\dot{\mathbf{q}} \tag{21}$$

where $\mathbf{J} = J_x^{-1}J_q$ is the Jacobian matrix of the robot.

The solution to the inverse kinematics problem is the solution to the redundancy problem.

2.4. Resolution of the Redundancy Problem

In the case of kinematic redundancy in parallel robots, the inverse kinematics problem has an infinite number of solutions. Solving the redundancy problem consists of obtaining a single solution to the inverse kinematics. One of the most common ways to obtain the inverse kinematic model is to use the following solution [12]:

$$\dot{\mathbf{q}} = \mathbf{J}^{\dagger}\dot{\mathbf{x}} - \gamma(\mathbf{I} - \mathbf{J}^{\dagger}\mathbf{J})\frac{\delta\mu}{\delta\mathbf{q}} \tag{22}$$

where $\mathbf{J}^{\dagger}$ is the Moore Penrose pseudo-inverse of $\mathbf{J}$, μ is a performance index to be minimized in order to perform a second task, and the matrix $\mathbf{I} - \mathbf{J}^{\dagger}\mathbf{J}$ maps to the null space of $\mathbf{J}$. The parameter γ is a scaling factor that changes the priority of the second task relative to the first. An example of such a secondary task is the minimizing of the following cost function:

$$\mu = \frac{1}{2}\sum_{i=1}^{n}\left(\frac{q_i - q_i^{mid}}{q_i^{max} - q_i^{min}}\right)^2 \tag{23}$$

where q_i^{mid} is the mid value that the i-th active joint can reach between q_i^{max} and q_i^{min}, which are the maximum and minimum values that the $i-th$ active joint can reach.

This cost function allows us to keep the active joints apart from their joint limit and thus ensure safe operations. Yet, there may be positions reachable by the unconstrained robot, but not by the robot with the actual joint limits. The gradient of (23) is given by

$$\frac{\delta \mu}{\delta \mathbf{q}} = \frac{q_i - q_i^{mid}}{\left(q_i^{max} - q_i^{min}\right)^2} \tag{24}$$

This solution for inverse kinematics, including the aforementioned secondary task, is used in the control laws to be defined.

2.5. Inverse Geometric Model

Two algorithms are proposed to solve the inverse geometric model to obtain a single solution. The first algorithm is based on the resolution of the inverse geometric model of the original delta robot, as it can be found in [16]. This algorithm consists of the following steps:

(i) Solve the inverse kinematics for a specified value of R as if it were the original delta robot.
(ii) Calculate the Jacobian matrix of the reconfigurable delta robot.
(iii) Calculate the condition number and compare it with the smallest previous condition number of the Jacobian matrix. If the current one is smaller, the actual joint values are the solution.
(iv) Repeat the process iteratively varying R over its entire range of motion, i.e., from 85 mm to 500 mm.
(v) The solution with the smallest condition number is obtained at the end of all iterations.

The advantages of this algorithm are that it is easy to implement and guarantees that the smallest minimum condition number is found. As a disadvantage, it has a high computational cost and may not be suitable for applications in real-time control laws. The second strategy to solve the inverse geometric model is to use an algorithm based on an inverse kinematic model in a numerical integration process. In such an algorithm, the desired value of **x** and the initial condition of the joints are provided, and at the end of the iterative process a solution for that **x** is obtained.

3. Image Jacobian

In this section, we present the development of a linear camera model based on the pinhole camera model [28], and develop an image Jacobian matrix that relates the velocities of the end effector in three-dimensional space to the velocities in camera space, from the derivative with respect to time of the linear camera model.

We start from the pinhole camera model shown in Figure 3 where we can see point P_i is projected onto the image plane through a segment passing via the origin of the camera coordinate frame (X, Y, Z). This way, we can relate one point in three-dimensional space to another in image space. To mathematically model such a projection, we first pose a homogeneous transformation from the global reference frame $(x,\ y,\ z)$ to the camera reference frame $(X,\ Y,\ Z)$ as follows:

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} R_{11} & R_{12} & R_{13} & X_0 \\ R_{21} & R_{22} & R_{23} & Y_0 \\ R_{31} & R_{32} & R_{33} & Z_0 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \mathbf{H} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \tag{25}$$

Assuming that the camera is fixed, the matrix **H** is constant. With the hole camera model, it is known that the projection of a point represented in the frame $(X,\ Y,\ Z)$ onto the frame $(x_c,\ y_c)$ in the image plane is as follows:

$$x_c = f\frac{X}{Z} \quad \text{y} \quad y_c = f\frac{Y}{Z}$$

where f is the focal length. The latter can be written in matrix form as:

$$\begin{bmatrix} x_c \\ y_c \\ 1 \end{bmatrix} = \frac{1}{Z} \begin{bmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \frac{1}{Z} \mathbf{F} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} \tag{26}$$

where

$$\mathbf{F} = \begin{bmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Assuming that the focal length is constant, then F is constant.

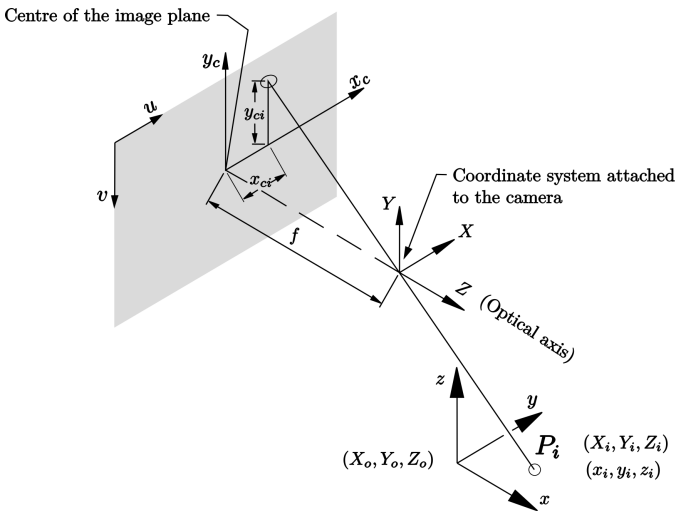

Figure 3. Pinhole camera model.

The coordinates in the frame $(x_c,\ y_c)$ are at the center of the image plane, and they are in length units. Another mapping is needed that moves the points to the frame typically used in digital imaging. That is, a (u, v) frame in the upper left corner of the image plane in which the direction of the u axis coincides with the direction of the x_c axis while the direction of the v axis is in the opposite direction to that of the y_c axis. In addition, we switch from longitudinal units to pixel units via the following homogeneous transformation:

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} m_x & 0 & p_x \\ 0 & -m_y & p_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_c \\ y_c \\ 1 \end{bmatrix} = \mathbf{K} \begin{bmatrix} x_c \\ y_c \\ 1 \end{bmatrix} \tag{27}$$

where $m_x,\ m_y$ represents the number of pixels per unit distance, and $p_x = c_x m_x$ and $p_y = c_y m_y$ are the coordinates of the center of the image plane in pixel units. All these parameters are considered constant for a camera so that the matrix $\mathbf{K}$ is constant.

We can perform the composition of all these homogeneous transformations to establish a direct transformation between the frame (x, y, z) and the frame (u, v) in the following way:

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \frac{1}{Z} \mathbf{KFH} \begin{bmatrix} x \\ y \\ z \end{bmatrix} \tag{28}$$

Developing the multiplication of matrices, we obtain

$$Z \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} c_{11} & c_{12} & c_{13} & c_{14} \\ c_{21} & c_{22} & c_{23} & c_{24} \\ c_{31} & c_{32} & c_{33} & c_{34} \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \tag{29}$$

where

$$c_{1j} = m_x(fR_{1j} + p_x R_{3j})$$
$$c_{14} = m_x(fX_0 + p_x Z_0)$$
$$c_{2j} = m_y(-fR_{1j} + p_y R_{3j})$$
$$c_{24} = m_y(-fX_0 + p_y Z_0)$$
$$c_{3j} = R_{3j}$$
$$c_{34} = Z_0$$

with $j = 1, 2, 3$.

Since this mapping is composed of eleven independent parameters, we can use one of the terms of the resulting matrix as a scaling factor and divide the equation by that element. The element taken is the last element of the matrix, i.e., Z_0. With this, we obtain the following expression:

$$\frac{Z}{Z_0}\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \frac{1}{Z_0}\begin{bmatrix} c_{11} & c_{12} & c_{13} & c_{14} \\ c_{21} & c_{22} & c_{23} & c_{24} \\ c_{31} & c_{32} & c_{33} & c_{34} \end{bmatrix}\begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \tag{30}$$

Which can be expressed as

$$\rho\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \mathcal{P}\begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \tag{31}$$

where

$$\mathcal{P} = \frac{1}{Z_0}\begin{bmatrix} c_{11} & c_{12} & c_{13} & c_{14} \\ c_{21} & c_{22} & c_{23} & c_{24} \\ c_{31} & c_{32} & c_{33} & c_{34} \end{bmatrix} = \begin{bmatrix} P_{11} & P_{12} & P_{13} & P_{14} \\ P_{21} & P_{22} & P_{23} & P_{24} \\ P_{31} & P_{32} & P_{33} & 1 \end{bmatrix} \tag{32}$$

and

$$\rho = \frac{Z}{Z_0} \tag{33}$$

From the third row of (31), another definition of ρ can be extracted as follows:

$$\rho = P_{31}x + P_{32}y + P_{33}z + 1 \tag{34}$$

Note that $\mathcal{P}$ in (32) is constant and independent of the depth scale factor ρ, thus obtaining as a result the following relationship, which we call the linear camera model:

$$\rho\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} P_{11} & P_{12} & P_{13} & P_{14} \\ P_{21} & P_{22} & P_{23} & P_{24} \\ P_{31} & P_{32} & P_{33} & 1 \end{bmatrix}\begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \tag{35}$$

From this result, the image Jacobian can be developed. To accomplish this, we can recast Equation (35) starting with developing the product of ρ with the image coordinates obtaining

$$u = \frac{P_{11}x + P_{12}y + P_{13}z + P_{14}}{P_{31}x + P_{32}y + P_{33}z + 1}$$

and

$$v = \frac{P_{21}x + P_{22}y + P_{23}z + P_{24}}{P_{31}x + P_{32}y + P_{33}z + 1}$$

which, when restructured in matrix form, provides:

$$\begin{bmatrix} u - P_{14} \\ v - P_{24} \end{bmatrix} = \begin{bmatrix} P_{11} - P_{31}u & P_{12} - P_{32}u & P_{13} - P_{33}u \\ P_{21} - P_{31}v & P_{22} - P_{32}v & P_{23} - P_{33}v \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} \tag{36}$$

or, in contracted form:

$$\mathbf{s}(t) = \mathbf{P}(\mathbf{s}(t))\,\mathbf{x}(t) \tag{37}$$

where

$$\mathbf{s}(t) = \begin{bmatrix} u - P_{14} \\ v - P_{24} \end{bmatrix},$$

and

$$\mathbf{P}(\mathbf{s}(t)) = \begin{bmatrix} P_{11} - P_{31}u & P_{12} - P_{32}u & P_{13} - P_{33}u \\ P_{21} - P_{31}v & P_{22} - P_{32}v & P_{23} - P_{33}v \end{bmatrix}.$$

The vectors $\mathbf{s}(t)$ and $\mathbf{x}(t)$ represent the coordinates in image space (subtracting (P_{14}, P_{24})) and Cartesian space, respectively. Both vectors are functions of time since they represent the position of a target that can move in three-dimensional space at any time. Deriving with respect to time Equation (37), we obtain

$$\dot{\mathbf{s}}(t) = \dot{\mathbf{P}}(\mathbf{s}(t))\mathbf{x}(t) + \mathbf{P}(\mathbf{s}(t))\dot{\mathbf{x}}(t) \tag{38}$$

where

$$\mathbf{x}(t) = \begin{bmatrix} x \\ y \\ z \end{bmatrix}$$

$$\dot{\mathbf{s}}(t) = \begin{bmatrix} \dot{u} \\ \dot{v} \end{bmatrix},$$

$$\dot{\mathbf{x}}(t) = \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \end{bmatrix},$$

$$\dot{\mathbf{P}}(\mathbf{s}(t)) = \begin{bmatrix} -P_{31}\dot{u} & -P_{32}\dot{u} & -P_{33}\dot{u} \\ -P_{31}\dot{v} & -P_{32}\dot{v} & -P_{33}\dot{v} \end{bmatrix}.$$

The term $\dot{\mathbf{P}}(\mathbf{s}(t))\mathbf{x}(t)$ in (38) can be developed as

$$\dot{\mathbf{P}}(\mathbf{s}(t))\mathbf{x}(t) = \begin{bmatrix} \dot{u}(-P_{31}x - P_{32}y - P_{33}z) \\ \dot{v}(-P_{31}x - P_{32}y - P_{33}z) \end{bmatrix}$$

$$\dot{\mathbf{P}}(\mathbf{s}(t))\mathbf{x}(t) = (-P_{31}x - P_{32}y - P_{33}z) \begin{bmatrix} \dot{u} \\ \dot{v} \end{bmatrix}$$

and finally

$$\dot{\mathbf{P}}(\mathbf{s}(t))\mathbf{x}(t) = (-\rho + 1)\dot{\mathbf{s}}(t). \tag{39}$$

Hence, we can restate (38) as follows:

$$\dot{\mathbf{s}}(t) = (-\rho + 1)\dot{\mathbf{s}}(t) + \mathbf{P}(\mathbf{s}(t))\dot{\mathbf{x}}(t)$$

$$\rho\dot{\mathbf{s}}(t) = \mathbf{P}(\mathbf{s}(t))\dot{\mathbf{x}}(t)$$

$$\dot{\mathbf{s}}(t) = \frac{1}{\rho}\mathbf{P}(\mathbf{s}(t))\dot{\mathbf{x}}(t)$$

$$\dot{\mathbf{s}}(t) = \mathbf{J}_{csm}(\mathbf{s}(t),\ \mathbf{x}(t))\dot{\mathbf{x}}(t) \tag{40}$$

where

$$\mathbf{J}_{csm}(\mathbf{s}(t),\ \mathbf{x}(t)) = \frac{1}{\rho(\mathbf{x}(t))}\mathbf{P}(\mathbf{s}(t)) \tag{41}$$

Thus, a direct relationship between the velocities in camera space and the velocities in three-dimensional space for each camera is obtained. This is of great value in the control context, since with this relation between velocities it is possible to define a closed-loop visual control laws using error signals defined as a function of the coordinates in image space. In order to establish an injective mapping in (40) a minimum of two cameras are required [6]. It can be observed that the image Jacobian $\mathbf{J}_{csm}$ in (41) is composed of the scale factor ρ and the mapping $\mathbf{P}$, which contains the vision parameters. In previous works [6,28,36], this ρ factor was assumed to be constant and equal to 1 (embedded in the vision parameters). While this approximation works locally, little is known about the size of the region of validity of such approximation. Moreover, the vision parameter needs to be calculated locally (refined) at each control iteration. In contrast, with the formulation presented here, the (constant) vision parameters need only to be determined once and the variable depth factor ρ can be obtained in several ways (for example by using the robot's geometric model, when the robot is close to the target; or by using 3D reconstruction from visual information. which can be as simple as a linear approximation). This not only reduces the computational cost of the calculation but also, and more importantly, makes the model globally valid, which is paramount from the automatic control point of view.

4. Control Laws for the Reconfigurable Robot

Usually, the sought behavior of most closed loop controlled systems is of the form

$$\dot{\mathbf{e}} = -\mathbf{G}\mathbf{e} \tag{42}$$

where $\mathbf{e}$ is the difference between a reference value and the current value of the system's state vector and $\mathbf{G}$ is a gain matrix. If $-\mathbf{G}$ is Hurwitz, an exponential convergence of $\mathbf{e}$ to zero is achieved. This behavior can be achieved for the system at hand by designing the following control laws, following the coordinates chosen to express the system behavior. Error $\mathbf{e}$ can be defined as $\mathbf{e} = \mathbf{q}^* - \mathbf{q}$, $\mathbf{e} = \mathbf{x}^* - \mathbf{x}$ and $\mathbf{e} = \mathbf{s}^* - \mathbf{s}$. That is, in joint space, operational (Cartesian) space or camera space, respectively.

On this basis, and considering the joint velocities as the control input, the following control law can be defined in joint space:

$$\dot{\mathbf{q}} = \mathbf{G}\mathbf{e} + \dot{\mathbf{q}}^* \tag{43}$$

where $\mathbf{e} = \mathbf{q}^* - \mathbf{q}$ and $\dot{\mathbf{q}}^*$ is the reference velocity in joint space. It is common to have the information of the joint values by using encoders with which actuators are usually controlled. Therefore, it is natural to think of a control with an error signal in the joint space, but it is not common to know the desired joint value. This information is not always available and represents an additional problem, which can be addressed by using the inverse kinematic model, which in turn requires the knowledge of the reference position in Cartesian space, usually not straightforwardly available, and solving the redundancy problem beforehand. This problem becomes even more difficult when it comes to obtaining the reference velocity. Furthermore, these calculations are subjected to the influence of model errors that are a known source of estimation error for these elements. This constitutes an obstacle for the implementation of this control law.

On the same basis, for the dynamical system expressed in operational (Cartesian) space

$$\dot{\mathbf{x}} = \mathbf{J}\dot{\mathbf{q}} \tag{44}$$

the following control law can be defined using the Jacobian and the solution of the reconfiguration problem:

$$\dot{\mathbf{q}} = \mathbf{J}^{\dagger}(\mathbf{G}\mathbf{e} + \dot{\mathbf{x}}^*) - \gamma(\mathbf{I} - \mathbf{J}^{\dagger}\mathbf{J})\frac{\delta\mu}{\delta\mathbf{q}} \tag{45}$$

where $\mathbf{e} = \mathbf{x}^* - \mathbf{x}$ and $\dot{\mathbf{x}}^*$ is the reference velocity in Cartesian space. In the case of a workspace error signal, it is common to calculate the end-effector position using the direct geometric model using the joint information. As for the target position, in general the desired position is also not known or requires some indirect measurement by using somewhat complex 3D reconstruction models, once again, sensible to errors in the models. Moreover, yet again, the tracking problem is difficult to carry out as they require real-time determination not only of the target position but of the target velocity. This constitutes an obstacle for the implementation of this control law.

Consider the dynamic system in camera space:

$$\dot{\mathbf{s}}(t) = \mathbf{J}_{csm}\mathbf{J}\dot{\mathbf{q}} \tag{46}$$

with the above conditions and using the mapping between the camera space and the operational space, the following control law can be defined:

$$\dot{\mathbf{q}} = \mathbf{J}^{\dagger}\mathbf{J}^{\dagger}{}_{csm}(\mathbf{G}\mathbf{e} + \dot{\mathbf{s}}^*) - \gamma(\mathbf{I} - \mathbf{J}^{\dagger}\mathbf{J})\frac{\delta\mu}{\delta\mathbf{q}} \tag{47}$$

where $\mathbf{e} = \mathbf{s}^* - \mathbf{s}$ and $\dot{\mathbf{s}}^*$ is the reference (target) velocity in camera space. Notice that both the reference position and the end-effector position in camera space are directly measured with the vision sensor and thus no model is directly involved to acquire them. As for the reference velocity, it can be estimated from the measurements of the reference position using a range of algorithms reported in the literature (e.g., Kalman filter, etc.). For these reasons, camera-space control can be adapted to a wide variety of practical applications.

All of the above control laws yield the same closed loop controlled system (42). In practice, one limitation of the above control laws is the exponential decay itself, which produce high initial control signals and, as the robot approaches the target, it decreases exponentially, slowing down the convergence. This behavior can be improved by implementing a variation of the exponential decay control law, using increasing gains as follows. In the model, we define the error dynamics as a piece-wise dynamic system where each partition has defined exponential decay dynamics. This can be written as follows:

$$\dot{\mathbf{e}} = \begin{cases} -\mathbf{G}_0\mathbf{e} & \text{if} \quad |\mathbf{e}| > d_1 \\ -\mathbf{G}_1\mathbf{e} & \text{if} \quad d_1 > |\mathbf{e}| > d_2 \\ -\mathbf{G}_2\mathbf{e} & \text{if} \quad d_2 > |\mathbf{e}| > d_3 \\ \qquad\vdots & \\ -\mathbf{G}_n\mathbf{e} & \text{if} \quad |\mathbf{e}| < d_n \end{cases} \tag{48}$$

where $\mathbf{G}_0 < \mathbf{G}_1 < \cdots < \mathbf{G}_n$ are matrices of the form $\mathbf{G} = \lambda\mathbf{I}$, where λ is any positive scalar, $d_1 > d_2 > ... > d_n > 0$ are the values of the error norm defining the partitions of the state space of the system.

In exponential decay control, as the value of $\mathbf{e}$ decreases, the value of $\dot{\mathbf{e}} = -\mathbf{G}\mathbf{e}$ decreases in the same proportion, causing a high time of convergence to the target point. In the piece-wise control variation, we seek to propose the gains $\mathbf{G_i}$ to increase in value as $\mathbf{e}$ decreases in value so that the rate of error dynamics approaches a linear decay rate, thus improving the speed of operation. If $\Delta d = |d_i - d_{i+1}|$ with $i = 1, 2, ..., n-1$, we can define $\mathbf{G}_i$ as

$$\mathbf{G}_0 = \lambda \cdot \mathbf{I} \tag{49}$$

and

$$\mathbf{G}_i = \prod_1^i \left(1 + \frac{\Delta d}{d_1 - (i-1)\Delta d}\right) \cdot \mathbf{G}_0 \tag{50}$$

Equation (50) guarantees that the eigenvalues of matrices $\mathbf{G}_i$ increase exponentially, compensating for the decrease in the value of the error norm. Since all matrices $-\mathbf{G}_i$ are Hurwitz in every partition whose union defines the entire state space, the dynamics of

system (48) is asymptotically stable. This is a variation of the control law for all of the above control laws.

4.1. Encoders Based Control Laws

Consider the dynamical system (44) where $\dot{\mathbf{q}}$ can be considered as the control input. Defining the closed-loop error dynamics (48) with $\mathbf{e} = \mathbf{x}^* - \mathbf{x}$ and the solution to the redundancy problem (22), we can obtain the following control law:

$$\dot{\mathbf{q}} = \begin{cases} \mathbf{J}^\dagger(\mathbf{G_0}(\mathbf{e}) + \dot{\mathbf{x}}^*) - \gamma(\mathbf{I} - \mathbf{J}^\dagger\mathbf{J})\frac{\delta\mu}{\delta\mathbf{q}} & \text{if} \quad |\mathbf{e}| > d_1 \\ \mathbf{J}^\dagger(\mathbf{G_1}(\mathbf{e}) + \dot{\mathbf{x}}^*) - \gamma(\mathbf{I} - \mathbf{J}^\dagger\mathbf{J})\frac{\delta\mu}{\delta\mathbf{q}} & \text{if} \quad d_1 > |\mathbf{e}| > d_2 \\ \mathbf{J}^\dagger(\mathbf{G_2}(\mathbf{e}) + \dot{\mathbf{x}}^*) - \gamma(\mathbf{I} - \mathbf{J}^\dagger\mathbf{J})\frac{\delta\mu}{\delta\mathbf{q}} & \text{if} \quad d_2 > |\mathbf{e}| > d_3 \\ \vdots & \\ \mathbf{J}^\dagger(\mathbf{G_n}(\mathbf{e}) + \dot{\mathbf{x}}^*) - \gamma(\mathbf{I} - \mathbf{J}^\dagger\mathbf{J})\frac{\delta\mu}{\delta\mathbf{q}} & \text{if} \quad |\mathbf{e}| < d_n \end{cases} \tag{51}$$

Another proposed control law is based on the closed-loop error dynamics (48) with the error defined in joint values: $\mathbf{e} = \mathbf{q}^* - \mathbf{q}$ obtaining the control law:

$$\dot{\mathbf{q}} = \begin{cases} \mathbf{G_0}(\mathbf{e}) + \dot{\mathbf{q}}^* & \text{if} \quad |\mathbf{e}| > d_1 \\ \mathbf{G_1}(\mathbf{e}) + \dot{\mathbf{q}}^* & \text{if} \quad d_1 > |\mathbf{e}| > d_2 \\ \mathbf{G_2}(\mathbf{e}) + \dot{\mathbf{q}}^* & \text{if} \quad d_2 > |\mathbf{e}| > d_3 \\ \vdots & \\ \mathbf{G_n}(\mathbf{e}) + \dot{\mathbf{q}}^* & \text{if} \quad |\mathbf{e}| < d_n \end{cases} \tag{52}$$

In this case, the redundancy problem is solved before the execution of the task to obtain the value of $\mathbf{q}^*$ using one of the algorithms presented in the previous section.

4.2. Control Law in Vision Sensor Space

Following the guidelines presented for developing control laws based on encoder signals and taking the dynamic system (46) where $\dot{\mathbf{q}}$ is the control input. The resulting control law is:

$$\dot{\mathbf{q}} = \begin{cases} \mathbf{J}^\dagger\mathbf{J}^\dagger_{\mathbf{csm}}(\mathbf{G_0}(\mathbf{e}) + \dot{\mathbf{s}}^*) - \gamma(\mathbf{I} - \mathbf{J}^\dagger\mathbf{J})\frac{\delta\mu}{\delta\mathbf{q}} & \text{if} \quad |\mathbf{e}| > d_1 \\ \mathbf{J}^\dagger\mathbf{J}^\dagger_{\mathbf{csm}}(\mathbf{G_1}(\mathbf{e}) + \dot{\mathbf{s}}^*) - \gamma(\mathbf{I} - \mathbf{J}^\dagger\mathbf{J})\frac{\delta\mu}{\delta\mathbf{q}} & \text{if} \quad d_1 > |\mathbf{e}| > d_2 \\ \mathbf{J}^\dagger\mathbf{J}^\dagger_{\mathbf{csm}}(\mathbf{G_2}(\mathbf{e}) + \dot{\mathbf{s}}^*) - \gamma(\mathbf{I} - \mathbf{J}^\dagger\mathbf{J})\frac{\delta\mu}{\delta\mathbf{q}} & \text{if} \quad d_2 > |\mathbf{e}| > d_3 \\ \vdots & \\ \mathbf{J}^\dagger\mathbf{J}^\dagger_{\mathbf{csm}}(\mathbf{G_n}(\mathbf{e}) + \dot{\mathbf{s}}^*) - \gamma(\mathbf{I} - \mathbf{J}^\dagger\mathbf{J})\frac{\delta\mu}{\delta\mathbf{q}} & \text{if} \quad |\mathbf{e}| < d_n \end{cases} \tag{53}$$

where $\dot{\mathbf{s}}^*$ is estimated using the numerical derivation of $\mathbf{s}^*$ from the measurements in the previous control cycles.

5. Experiments

This section presents the experimental setup implemented to validate the control laws designed in the previous sections. First, a brief description of the hardware and software will be given of the different parts of the test bench used.

5.1. Hardware

We used a reconfigurable variant of the delta PARALLIX LKF-2040 robot. The kinematic description of the robot is presented in Section 2. The robot is composed of four maxon 380961 motors, each of them controlled with a PICSERVO SC board. Communication with the control boards is serial over USB. The boards control the actuators using an internal PID controller and receive speeds as commands. A conveyor has been placed inside the workspace of the robot to allow for the tracking of moving objects.

The vision system is composed of two Allied Vision cameras of the Alvium line at 1296 × 964 resolution capturing at 130 frames per second through UBS 3.0. The cameras are located at a distance of approximately 1.5 m from the robot. Each camera is 1 m apart. A personal computer with Intel(R) Core(TM) i7-8700 CPU @ 3.20GHz of 6 and 12 cores, 16 gigabytes of RAM, was used to control the robot controller boards, the vision system, and to perform the control law calculation. Figure 4 shows a picture of the full system and Figure 5 shows an schematic of the hardware system and its workflow.

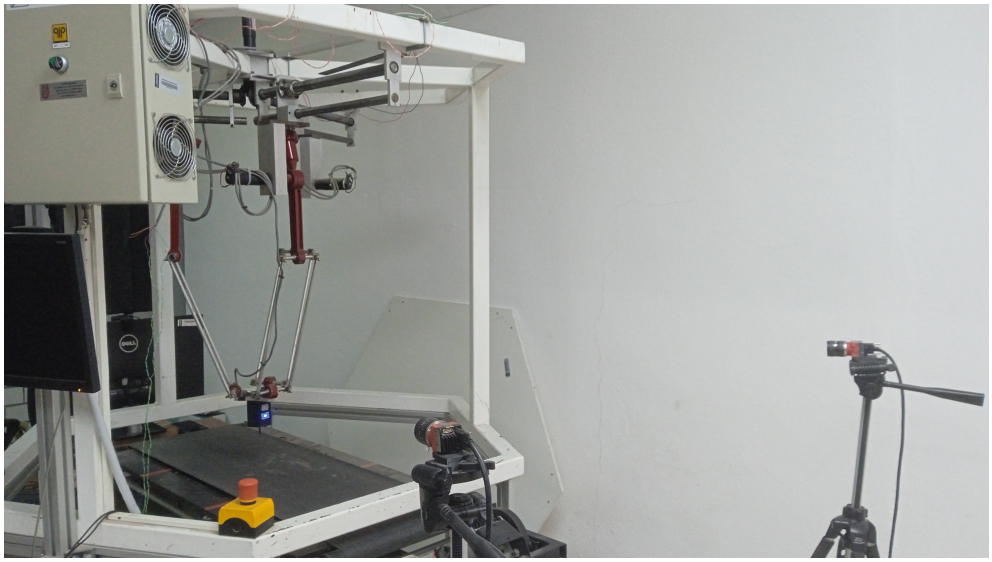

Figure 4. The reconfigurable delta robot and camera system.

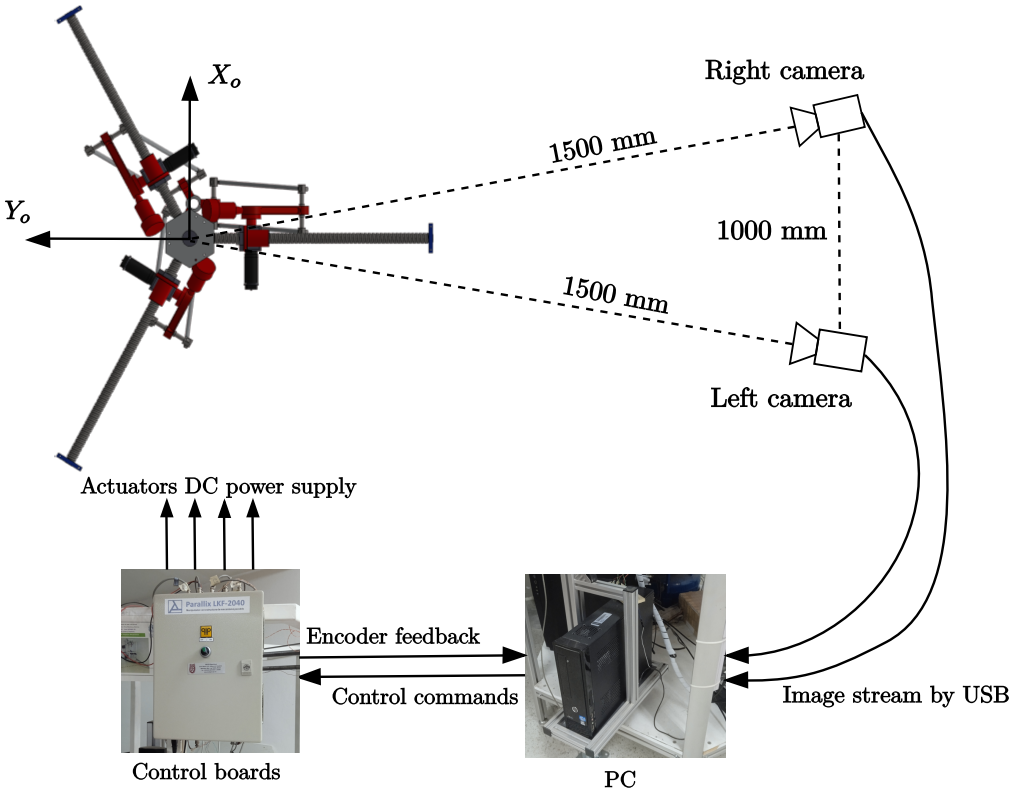

Figure 5. Schematic of the hardware system and its workflow.

5.2. Software

The control system was programmed using Python 3.10. Numerical computations, necessary in the control law, were performed using NumPy 1.24.3 [37]. Image capture was handled via VimbaPython 6.0 SDK provided by the camera manufacturer while image processing used OpenCV 4.7 [38]. Finally, the PyQt 5.15.9 [39] library was used to generate a graphical interface and thread management. The program is composed of three main threads:

(i) A main thread for all the cameras,
(ii) A thread to send commands to the robot,
(iii) A thread for the control law calculation.

The control cycle time obtained with the aforementioned set-up was 16 ms.

5.3. Testing Configuration

First, static positioning tasks are used to validate and compare the performance of exponential convergence control laws ((45), (43) and (47)) versus the piece-wise variants ((51), (52) and (53)) with reference velocity equal zero. Reference positions were distributed in the robot workspace and the robot was commanded to reach those positions using the designed control laws. For the vision-based control law, the vision parameters were initially estimated using what has been referred to as a pre-plan. During this procedure, the robot is positioned at a series of points covering its workspace. For these tests, 410 measurements were used, evenly distributed throughout the robot's workspace, with 50 mm spacing on the X and Y axes and 25 mm spacing on the Z axis.

Second, in order to fully validate the vision-based control law (53) and evaluate its performance, two types of experiments are considered:

(i) The first set of experiments for the CSM control law consists of static positioning tasks. Four position tasks are performed three times each. The positions are chosen randomly inside the workspace of the robot. The maneuver is considered successful when an error vector norm of less than 2 pixel is achieved during, at least, 40 control cycles. This error vector contains the coordinates in image space of the two cameras. Speed compensation is set to zero to avoid noise input due to the velocity estimator.
(ii) The second set of experiments for the CSM control law consists of tracking an object moving, on the conveyor, following a linear trajectory at constant speed. No a priori information about the target trajectory is used in the experiments. The target is positioned on a conveyor belt placed within the robot's workspace. The robot first moves to the target object. The conveyor is started only once a static position has been achieved using the previously mentioned conditions. The conveyor moves at a constant speed with the robot following the object's position. The test is stopped when the object reaches the edge of the robot's workspace. The control law with speed compensation is used.

Figure 6 shows the general operation of the control system using the example of visual control for positioning tasks, where the stop criterion corresponds to when the error norm is less than 2 pixels. The operation is similar for the tracking task, with the difference that the stop criterion corresponds to the target leaving the assigned tasking area.

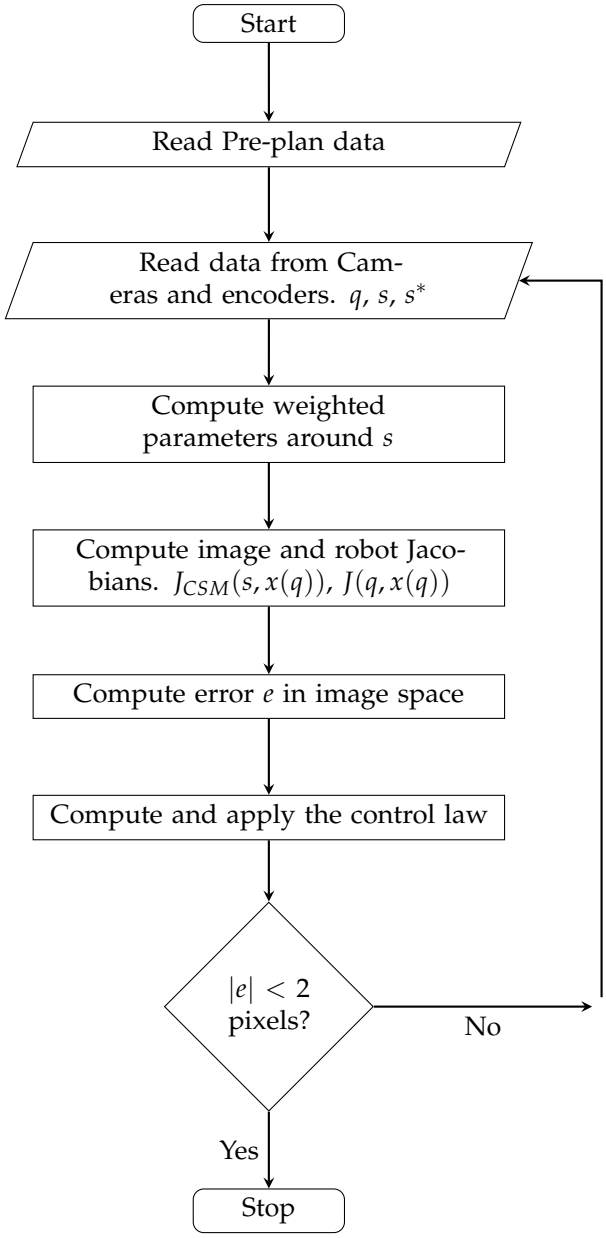

Figure 6. General operation of the control system.

6. Results and Discussion

6.1. Results

6.1.1. Exponential Convergence Control Laws vs. Piece-Wise Variants

Static positioning tasks using exponential convergence control laws ((43), (45) and (47)) were executed to compare their performance with the corresponding piece-wise variants ((51)–(53)). Only results for the control laws in the operational space are shown, although similar results were obtained for the control laws in the other spaces. Figures 7 and 8 show the error norm and control signal, respectively, for control law (45). Similarly, Figures 9 and 10 show the error norm and control signals for the piece-wise variant (51). Comparing the images, which consist of the same positioning task, it can be seen how the piece-wise variant is effective in improving the convergence time from 4 s to 1.8 s, while preserving the steady state zero error of the exponential convergence control law.

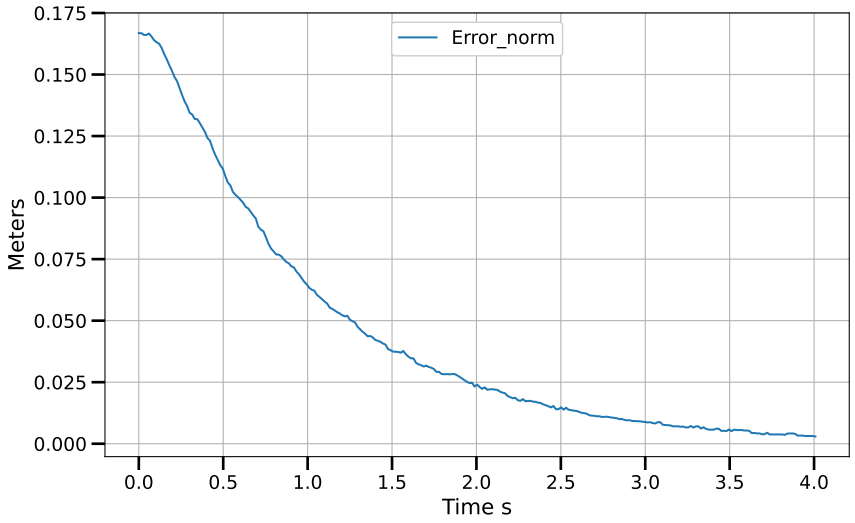

Figure 7. Error norm vs. time for control law $\dot{\mathbf{q}} = \mathbf{J}^{\dagger}\mathbf{G}(\mathbf{e}) - \gamma(\mathbf{I} - \mathbf{J}^{\dagger}\mathbf{J})\frac{\delta\mu}{\delta\mathbf{q}}$ with $\mathbf{e} = \mathbf{x}^{*} - \mathbf{x}$.

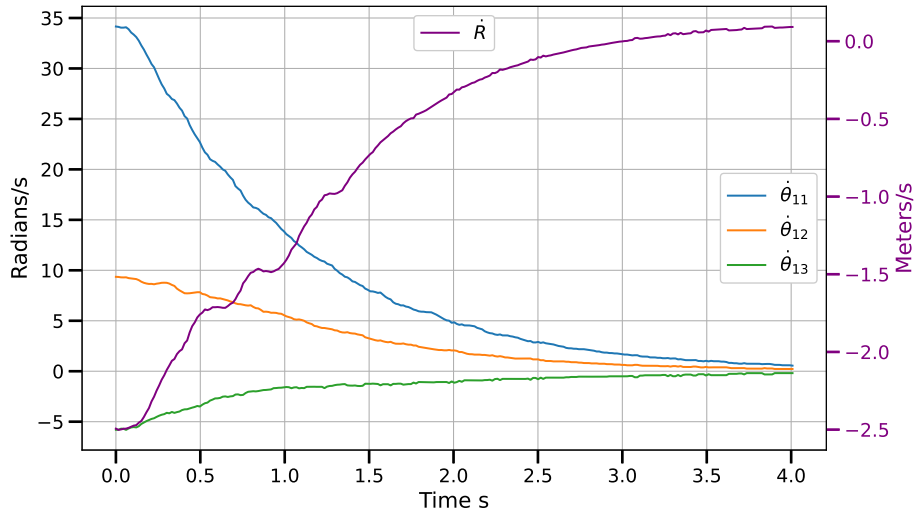

Figure 8. Angular joint velocity (radians/s) vs. time for control law $\dot{\mathbf{q}} = \mathbf{J}^{\dagger}\mathbf{G}(\mathbf{e}) - \gamma(\mathbf{I} - \mathbf{J}^{\dagger}\mathbf{J})\frac{\delta\mu}{\delta\mathbf{q}}$ with $\mathbf{e} = \mathbf{x}^{*} - \mathbf{x}$.

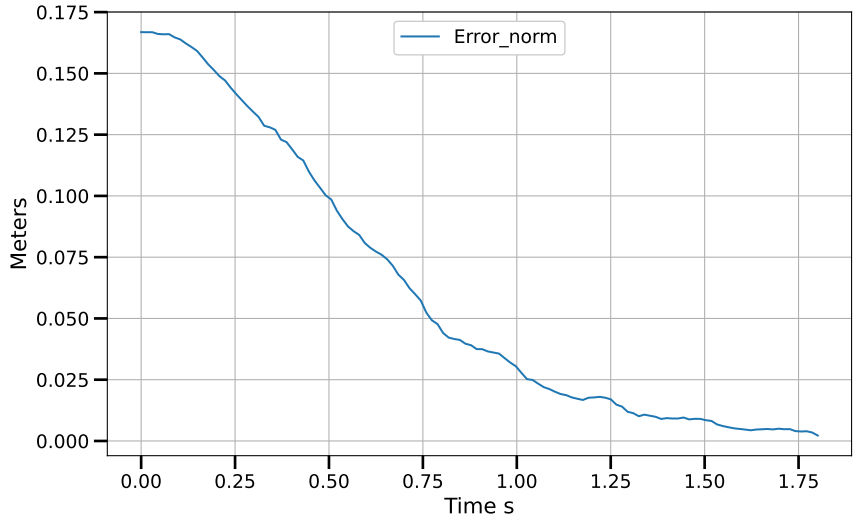

Figure 9. Error norm vs. time for control law (51) with $\mathbf{e} = \mathbf{x}^{*} - \mathbf{x}$.

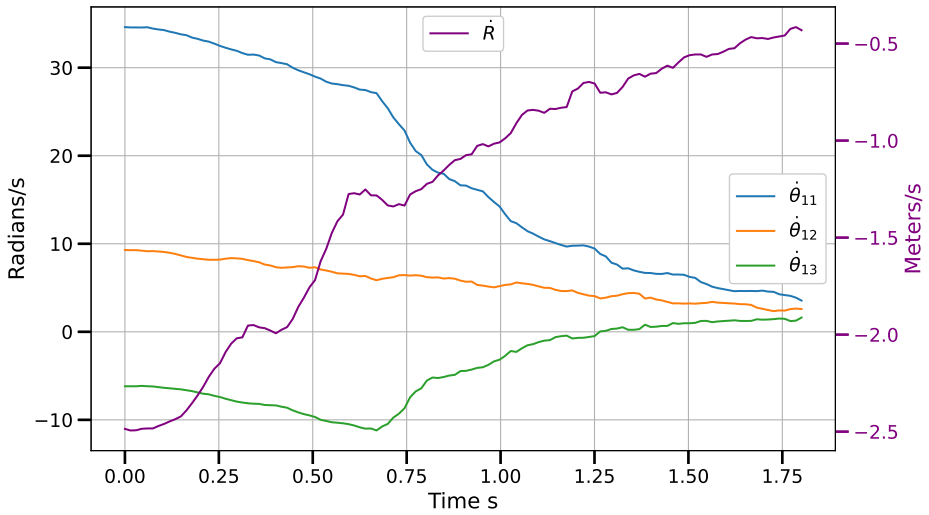

Figure 10. Angular joint velocity (radians/s) vs. time for control law (51) with $\mathbf{e} = \mathbf{x}^* - \mathbf{x}$.

6.1.2. Control Law in Vision-Sensor Space

For the first set of experiments (i.e., static positioning) for the vision-based control law, an average error of 0.604 mm was obtained. Table 1 shows more detail about these results. The maximum and minimum errors coincide with the target point farthest from the cameras and the target point closest to the cameras. This can happen because the pixel/mm ratio decreases as the distance increases (this phenomenon is also observed in the tracking tasks reported below).

Figure 11 shows the position error norm in image coordinates, measured in pixels. It can be seen how the behavior of the norm is similar to a linear decay in most of the robot's path. This is a direct effect of the piece-wise control law. Figure 12 shows the velocity commands sent to the control boards. The angular velocities ($\dot{\theta}_{1i}$) of the actuated rotational joints are measured on the scale to the left, while the velocity of the prismatic reconfiguration joint ($\dot{R}$) is measured on the scale to the right. In all cases, the signals tend to zero.

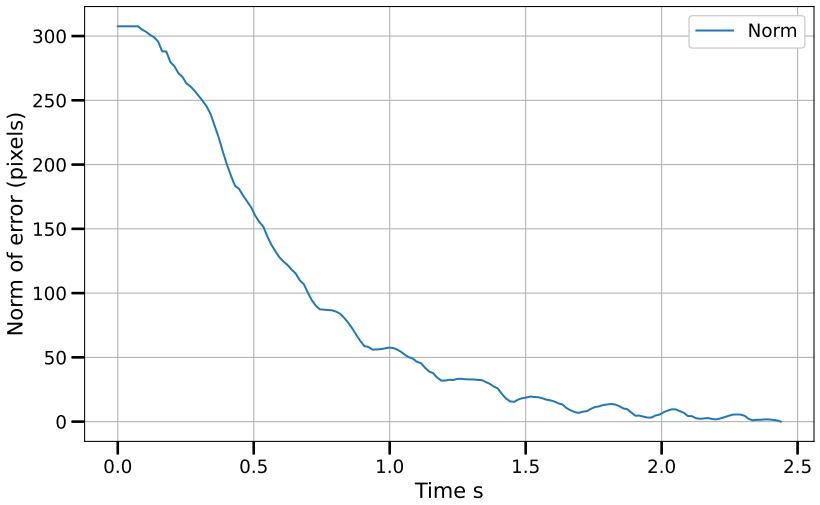

Figure 11. Positioning error norm in image space.

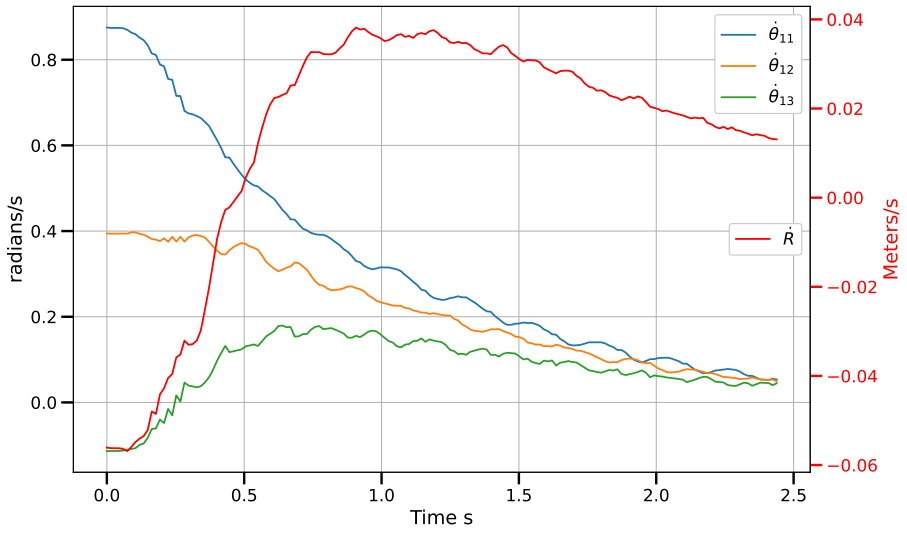

Figure 12. Velocity control signal.

Table 1. Position error for positioning task.

	Error (mm)
Average	0.600
Max	1.530
Min	0.172
Std. Dev.	0.445

For the tracking task of the CSM control law, for an object moving at constant speed, average drag errors were measured between 2.5 mm and 11.5 mm for speeds ranging from 3.5 cm/s to 12.7 cm/s. The tracking error increases as the belt speed increases. The results are shown in the Table 2. Figures 13–15 show the tracking error in both the image and operational spaces, for the experiments at different conveyor belt speeds. The tracking error in the image space is represented by the blue line and is measured on the left hand scale while the tracking error in the operational space is represented by the violet line and is presented on the right hand scale. At the beginning of the experiments at 3.5 cm/s and 9.3 cm/s, the object is motionless and the robot positions on the target; then, the object starts moving at a constant speed and the robot tracks the target. For the experiment at 12.7 cm/s, the target starts moving from the beginning of the experiment. Table 2 shows the tracking errors in operational space. The tracking error consists of the average of the error norm from the time the target starts its movement to the end of the experiment when the target leaves the working area. This time can vary depending on the tracking speed between 2.5 s and 5 s. The working area allows for a trajectory of the object covering a distance of 38 cm. We can see that the average error is always a small fraction of the distance covered by the target and stays always below 3%.

Table 2. Tracking error in operational space for constant velocity task.

Conveyor Speed	**Average Error**
6.5 cm/s	2.5 mm
9.3 cm/s	3.9 mm
12.7 cm/s	11.5 mm

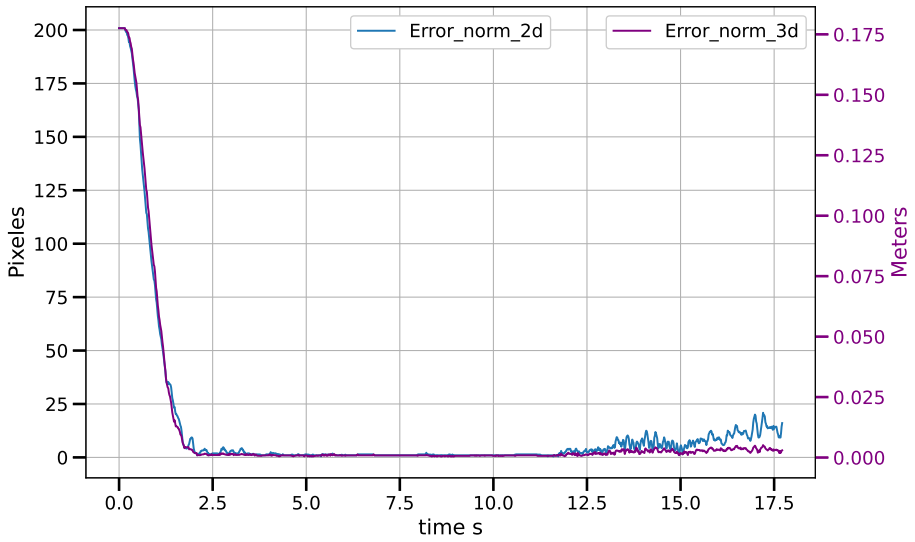

Figure 13. Tracking error in image and operational spaces for a speed of 6.5 cm/s.

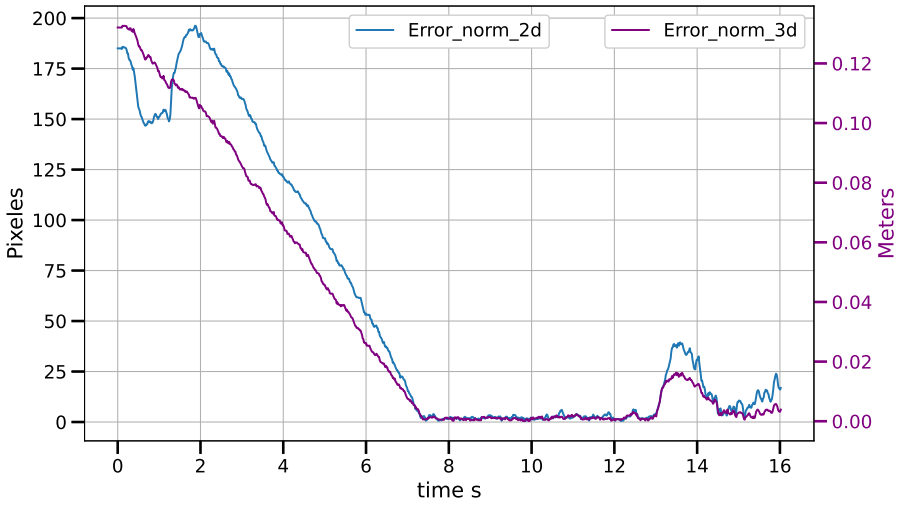

Figure 14. Tracking error in image and operational spaces for a speed of 9.3 cm/s.

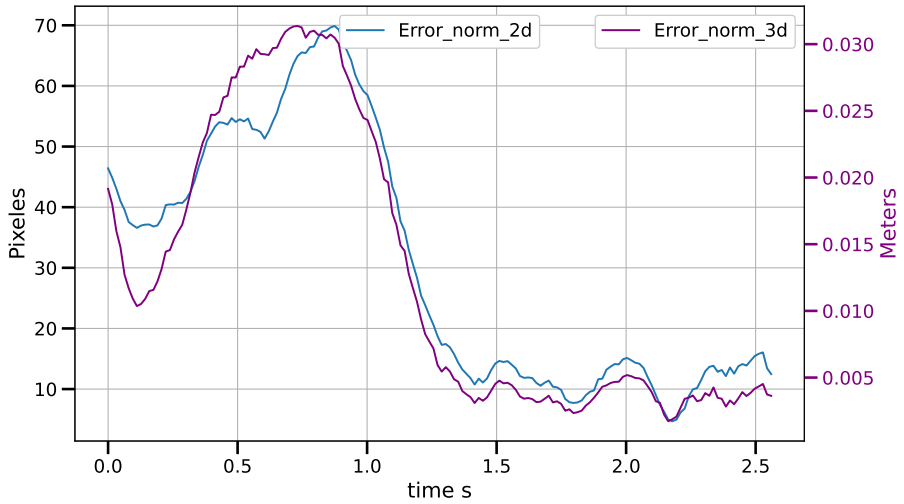

Figure 15. Tracking error in image and operational spaces for a speed of 12.7 cm/s.

6.2. Discussion

In the comparison between exponential convergence control laws and piece-wise variants, it can be seen how the piece-wise variant is effective in improving the convergence time to about half the original time, while preserving the steady state zero error of the exponential convergence control law. In practical applications, this represents a significant advantage, as it can accelerate processes.

With respect to the control laws designed in the vision sensor-space, in static positioning tasks, an average positioning error of 0.6 mm was obtained. This implies an improvement in positioning precision over previous implementations of LCM-CSM/CSM control laws [6,28,36] where an accuracy of about 1 mm was achieved. This improvement can be attributed both to the implementation of the LCM-CSM Jacobian matrix that includes the depth scale information and to the higher resolution of the cameras. In the tests performed, a loss of accuracy in the positioning task was observed as the target moves away from the cameras. This can be explained by the fact that the farther away the cameras, the lower the pixel/mm correspondence.

For the constant velocity trajectory tracking task, tracking errors of 2.5 mm to 5 mm were achieved for velocities of 6.5 to 12.7 cm/s. These results are competitive with other results reported in the literature. Similar to the positioning task, there is an increase in error as the target moves away from the cameras. It was found that the velocities estimated by averaging the last 10 numerical derivatives produce very noisy signals that are transferred to the control signal. This can affect the performance of the robot and generate a higher tracking error compared to a cleaner velocity signal.If you have a polynomial estimate of the trajectory in the last 10 steps, the velocity can be calculated analytically.

7. Conclusions

In this article, the Jacobian analysis and the design of vision-based control laws for a reconfigurable, redundant delta-type parallel robot are presented. We investigated a target-tracking task where the target trajectory is not known in advance. Also, the image Jacobian (J_{csm}) based on the LCM-CSM approach is revisited and the vision parameters are shown to be constant and independent of a depth scale factor, the function of the depth of the observed features, which is variable. This allows us to very easily update the image Jacobian matrix at each control cycle with a very low computational cost.

Using the proposed control strategy, in the case of static objectives, the average positioning error was 0.6 mm with a standard deviation of 0.445 mm. These results are competitive when compared against a number of industrial applications. Regarding mobile object tracking tasks, the results show a tracking errors of 2.5 mm, 3.9 mm and 5 mm for targets moving along a linear trajectory at speeds of 6.5, 9.3 and 12.7 cm/s, respectively, with a maximum final static positioning errors of 1.53 mm once the object stopped.

Parallel robots require specialized control designs because of their model complexity and open problems, which prevent simply extending existing robot manipulator controls. Moreover, while a reconfigurable parallel manipulator increases the versatility of the manipulator and tackles some of its limitations, it also increases the complexity of the robot and requires an even more specialized control. Delta robots are very popular in industry, particularly in "pick and place" tasks; however, these applications require online adjustments, specification changes and the tracking of moving targets. Under these circumstances, vision-based control provides the necessary flexibility over traditional control approaches and, in the authors' opinion, the proposed vision-based control via CSM provides a satisfactory solution for said tasks.

Future Work

The results presented in this article are a starting point for the topics addressed here. On the one hand, it is possible to deepen the analysis of the characteristics resulting from the reconfiguration of the robot. On the other hand, the presented piecewise control law shows a significant improvement in the convergence rate compared to the exponential decay

control law. Comparisons with other control laws, such as PID, which are traditionally one of the first choices for controlling a system, can shed light on the full potential of the proposed law. The development of a control methodology around the proposed law is desirable. Considering the VLCM-CSM visual control, comparisons can be made between the Jacobian in this article with those reported in other papers to obtain a better sense of its effectiveness.

Author Contributions: Conceptualization, A.F.-L., M.M., A.G. and A.C.; Methodology, A.F.-L., M.M. and D.P.; Software, A.F.-L.; Validation, A.F.-L.; Formal analysis, A.F.-L., M.M. and D.P.; Investigation, M.M., A.G., A.C. and D.P.; Resources, M.M.; Data curation, A.G.; Writing—original draft, A.F.-L.; Writing—review & editing, A.F.-L., M.M., A.G., A.C. and D.P.; Visualization, A.C.; Supervision, M.M.; Project administration, M.M.; Funding acquisition, M.M. All authors have read and agreed to the published version of the manuscript.

Funding: This work was partially funded by CONACyT grant Cátedras CONACyT 2016/972 and CONHACyT grant 712819. The APC was funded by Posgrado en Ing. Mecánica, Posgrado en Ing. Eléctrica, and Ing. en Mecatrónica, Facultad de Ingeniería, UASLP.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Data available upon reasonable request.

Conflicts of Interest: The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

1. Clavel, R. DELTA, a fast robot with parallel geometry. In Proceedings of the 18th International Symposium on Industrial Robots, Lausanne, Switzerland, 26–28 April 1988; Burckhardt, C.W., Ed.; Springer: New York, NY, USA, 1988; pp. 91–100.
2. Merlet, J.P. *Parallel Robots*; Kluwer Academic Publishers: Alphen aan den Rijn, The Netherlands, 2006.
3. Mostashiri, N.; Dhupia, J.S.; Verl, A.W.; Xu, W. A Review of Research Aspects of Redundantly Actuated Parallel Robotsw for Enabling Further Applications. *IEEE/ASME Trans. Mechatronics* **2018**, *23*, 1259–1269. [CrossRef]
4. Pandilov, Z.; Dukovski, V. Comparison of the characteristics between serial and parallel robots. *Acta Tech. Corviniensis-Bull. Eng.* **2014**, *7*, 143–160.
5. Zhang, H.; Fang, H.; Zou, Q. Non-singular terminal sliding mode control for redundantly actuated parallel mechanism. *Int. J. Adv. Robot. Syst.* **2020**, *17*, 1729881420919548. [CrossRef]
6. Loredo, A.; Maya, M.; González, A.; Cardenas, A.; Gonzalez-Galvan, E.; Piovesan, D. A Novel Velocity-Based Control in a Sensor Space for Parallel Manipulators. *Sensors* **2022**, *22*, 7323. [CrossRef] [PubMed]
7. Pandilov, Z.; Dukovski, V. Several open problems in parallel robotics. *Acta Tech. Corviniensis-Bull. Eng.* **2011**, *4*, 77–84.
8. Bentaleb, T.; Iqbal, J. On the improvement of calibration accuracy of parallel robots—Modeling and optimization. *J. Theor. Appl. Mech.* **2020**, *58*, 261–272. [CrossRef]
9. Milutinović, D.; Slavković, N.; Kokotović, B.; Milutinović, M.; Živanović, S.; Dimić, Z. Kinematic modeling of reconfigurable parallel robots based on DELTA concept. In Proceedings of the 11th International Scientific Conference MMA 2012—Advanced Production Technologies, Novi Sad, Serbia, 20–21 September 2012; University of Novi Sad, Faculty of Technical Scienes, Department for Production Engineering: Novi Sad, Serbia, 2012; pp. 262–259.
10. Plitea, N.; Lese, D.; Pisla, D.; Vaida, C. Structural design and kinematics of a new parallel reconfigurable robot. *Robot. Comput.-Integr. Manuf.* **2013**, *29*, 219–235. [CrossRef]
11. Luces, M.; Beno Benhabib, J.K.M. A Review of Redundant Parallel Kinematic Mechanisms. *J. Intell. Robot. Syst.* **2017**, *86*, 175–198. [CrossRef]
12. Gosselin, C.; Schreiber, L.T. Redundancy in Parallel Mechanisms: A Review. *Appl. Mech. Rev.* **2018**, *70*, 010802. [CrossRef]
13. Mostashiri, N.; Dhupia, J.; Xu, W. Redundancy in Parallel Robots: A Case Study of Kinematics of a Redundantly Actuated Parallel Chewing Robot. In Proceedings of the RITA 2018, Kuala Lumpur, Malaysia, 16–18 December 2018; Abdul Majeed, A.P.P., Mat-Jizat, J.A., Hassan, M.H.A., Taha, Z., Choi, H.L., Kim, J., Eds.; Springer: Singapore, 2020; pp. 65–78.
14. Harada, T.; Angeles, J. From the McGill pepper-mill carrier to the Kindai ATARIGI Carrier: A novel two limbs six-dof parallel robot with kinematic and actuation redundancy. In Proceedings of the 2017 IEEE International Conference on Robotics and Biomimetics (ROBIO), Macau, China, 5–8 December 2017; pp. 1328–1333. [CrossRef]
15. Moosavian, A.; Xi, F.J. Modular design of parallel robots with static redundancy. *Mech. Mach. Theory* **2016**, *96*, 26–37. [CrossRef]
16. Maya, M.; Castillo, E.; Lomelí, A.; González-Galván, E.; Cárdenas, A. Workspace and Payload-Capacity of a New Reconfigurable Delta Parallel Robot. *Int. J. Adv. Robot. Syst.* **2013**, *10*, 56, [CrossRef]

17. Hu, L.; Gao, H.; Qu, H.; Liu, Z. Closeness to singularity based on kinematics and dynamics and singularity avoidance of a planar parallel robot with kinematic redundancy. *Proc. Inst. Mech. Eng. Part C: J. Mech. Eng. Sci.* **2022**, *236*, 3717–3730, [CrossRef]
18. Pulloquinga, J.L.; Escarabajal, R.J.; Ferrándiz, J.; Vallés, M.; Mata, V.; Urízar, M. Vision-Based Hybrid Controller to Release a 4-DOF Parallel Robot from a Type II Singularity. *Sensors* **2021**, *21*, 4080. [CrossRef]
19. Singh, A.; Kalaichelvi, V.; Karthikeyan, R. A survey on vision guided robotic systems with intelligent control strategies for autonomous tasks. *Cogent Eng.* **2022**, *9*, 2050020. [CrossRef]
20. Cherubini, A.; Navarro-Alarcon, D. Sensor-based control for collaborative robots: Fundamentals, challenges, and opportunities. *Front. Neurorobot.* **2021**, *14*, 113. [CrossRef] [PubMed]
21. Hutchinson, S.; Hager, G.D.; Corke, P.I. A tutorial on visual servo control. *IEEE Trans. Robot. Autom.* **1996**, *12*, 651–670. [CrossRef]
22. Chaumette, F.; Hutchinson, S. Visual servo control. I. Basic approaches. *IEEE Robot. Autom. Mag.* **2006**, *13*, 82–90. [CrossRef]
23. Ren, X.; LI, H. Uncalibrated Image-Based Visual Servoing Control with Maximum Correntropy Kalman Filter. *IFAC-PapersOnLine* **2020**, *53*, 560–565. [CrossRef]
24. Skaar, S.B.; Brockman, W.H.; Hanson, R. Camera-Space Manipulation. *Int. J. Robot. Res.* **1987**, *6*, 20–32. [CrossRef]
25. Xie, Y.; Wang, W.; Zhao, C.; Skaar, S.B.; Wang, Q. A Differential Evolution Approach for Camera Space Manipulation. In Proceedings of the 2020 2nd World Symposium on Artificial Intelligence (WSAI), Karlsruhe, Germany, 14–16 September 2020; IEEE: Manhattan, NY, USA, 2020; pp. 103–107.
26. Weiss, L.; Sanderson, A.; Neuman, C. Dynamic sensor-based control of robots with visual feedback. *IEEE J. Robot. Autom.* **1987**, *3*, 404–417. [CrossRef]
27. Lin, C.J.; Shaw, J.; Tsou, P.C.; Liu, C.C. Vision servo based Delta robot to pick-and-place moving parts. In Proceedings of the 2016 IEEE International Conference on Industrial Technology (ICIT), Taipei, Taiwan, 14–17 March 2016; pp. 1626–1631. [CrossRef]
28. Rendón-Mancha, J.M.; Cárdenas, A.; García, M.A.; Lara, B. Robot positioning using camera-space manipulation with a linear camera model. *IEEE Trans. Robot.* **2010**, *26*, 726–733. [CrossRef]
29. Huynh, B.P.; Kuo, Y.L. Dynamic Hybrid Filter for Vision-Based Pose Estimation of a Hexa Parallel Robot. *J. Sens.* **2021**, *2021*, 9990403. [CrossRef]
30. Amjad, J.; Humaidi, A.I.A. Design of Augmented Nonlinear PD Controller of Delta/Par4-Like Robot. *J. Control Sci. Eng.* **2019**, *2019*, 7689673. [CrossRef]
31. Hoang, X.B.; Pham, P.C.; Kuo, Y.L. Collision Detection of a HEXA Parallel Robot Based on Dynamic Model and a Multi-Dual Depth Camera System. *Sensors* **2022**, *22*, 5922. [CrossRef] [PubMed]
32. Kansal, S.; Mukherjee, S. Vision-based kinematic analysis of the Delta robot for object catching. *Robotica* **2022**, *40*, 2010–2030. [CrossRef]
33. Huynh, B.P.; Kuo, Y.L. Dynamic Filtered Path Tracking Control for a 3RRR Robot Using Optimal Recursive Path Planning and Vision-Based Pose Estimation. *IEEE Access* **2020**, *8*, 174736–174750. [CrossRef]
34. Zake, Z.; Chaumette, F.; Pedemonte, N.; Caro, S. Vision-Based Control and Stability Analysis of a Cable-Driven Parallel Robot. *IEEE Robot. Autom. Lett.* **2019**, *4*, 1029–1036. [CrossRef]
35. Zake, Z.; Chaumette, F.; Pedemonte, N.; Caro, S. Robust 2½D Visual Servoing of A Cable-Driven Parallel Robot Thanks to Trajectory Tracking. *IEEE Robot. Autom. Lett.* **2020**, *5*, 660–667. [CrossRef]
36. Lopez-Lara, J.G.; Maya, M.E.; González, A.; Cardenas, A.; Felix, L. Image-based control of delta parallel robots via enhanced LCM-CSM to track moving objects. *Ind. Robot. Int. J. Robot. Res. Appl.* **2020**, *47*, 559–567. [CrossRef]
37. Harris, C.R.; Millman, K.J.; van der Walt, S.J.; Gommers, R.; Virtanen, P.; Cournapeau, D.; Wieser, E.; Taylor, J.; Berg, S.; Smith, N.J.; et al. Array programming with NumPy. *Nature* **2020**, *585*, 357–362. [CrossRef]
38. Bradski, G. The OpenCV Library. *Dr. Dobb'S J. Softw. Tools Prof. Program.* **2000**, *25*, 120–123.
39. PyQt. PyQt Reference Guide. 2012. Available online: https://doc.bccnsoft.com/docs/PyQt5/ (accessed on 7 June 2023).