Image-based Visual Servoing of a Gough-Stewart Parallel Manipulator using Leg Observations

Nicolas Andreff, Tej Dallej, Philippe Martinet

Image-based Visual Servoing of Gough-Stewart Parallel Manipulators using Legs Observation

Nicolas Andreff* Tej Dallej* Philippe Martinet*

June 12, 2006

Abstract

In this paper, a tight coupling between computer vision and parallel robotics is exhibited through the projective line geometry. Indeed, contrary to the usual methodology where the robot is modeled independently from the control law which will be implemented, we take into account, since the early modeling stage, that vision will be used for control. Hence, kinematic modeling and projective geometry are fused into a control-devoted projective kinematic model. Thus, a novel vision-based kinematic modeling of a Gough-Stewart manipulator is proposed through the image projection of its cylindrical legs. Using this model, a visual servoing scheme is presented, where the image projection of the non-rigidly linked legs are servoed, rather than the end-effector pose.

1 Introduction

Parallel mechanisms are such that there exist several kinematic chains (or legs) between their base and their end-effector. Therefore, they may exhibit a better repeatability [30] than serial mechanisms but not a better accuracy [38], because of the large number of links and passive joints. There can be two ways to compensate for the low accuracy. The first way is to perform a kinematic calibration of the mechanism and the second one is to use a control law which is robust to calibration errors.

There exists a large amount of work on the control of parallel mechanisms (see [29] for a long list of references). In the focus of attention, Cartesian control is naturally achieved through the use of the inverse differential kinematic model (abusively called the robot Jacobian since velocities in the Cartesian space do not form a vector space) which transforms Cartesian velocities into joint velocities. It is noticeable that the inverse differential kinematic model of parallel mechanisms does not only depend on the joint configuration (as for serial mechanisms) but also on the end-effector pose.

*LASMEA-UMR 6602, CNRS-Université Blaise Pascal, 24 avenue des Landais, F-63177 Aubière Cedex, Email: firstname.lastname@lasmea.univ-bpclermont.fr, Web page: http://www.lasmea.univ-bpclermont.fr/Control

Figure 1: A Gough-Stewart platform.

Consequently, one needs to be able to estimate or measure the latter. As far as we know, all the effort has been put on the estimation of the end-effector pose through the forward kinematic model and the joint measurements. However, this yields much trouble, related to the fact that there is usually no analytic formulation of the forward kinematic model of a parallel mechanism. Hence, one numerically inverts the inverse kinematic model, which is analytically defined for most of the parallel mechanisms. However, it is known [28, 16] that this numerical inversion requires high order polynomial root determination, with several possible solutions (up to 40 real solutions [10] for a Gough-Stewart platform [14, 37]). Much of the work is thus devoted to solving this problem accurately and in real-time (see for instance [39]), or to designing parallel mechanisms with analytical forward kinematic model [18, 13]. Alternately, one of the promising paths lies in the use of the so-called metrological redundancy [6], which simplifies the kinematic models by introducing additional sensors into the mechanism and thus yields easier control [26].

Vision being an efficient way of estimating the end-effector pose [9, 21], it is a good alternative to use it for Cartesian control of parallel mechanisms. It can be done in three ways.

Vision as a sensor The first one consists in computing the end-effector poses by vision, then in translating them into joint configurations, through the inverse kinematic model, and finally servoing in the joint space. This scheme is rather easy to implement for serial mechanisms provided that inverting the forward kinematic model can be done satisfactorily. The latter is straightforward for parallel mechanisms since they usually have an analytical inverse kinematic model. Similarly, one can consider computer vision as a contact-less redundant sensor, as already stated in the context of parallel mechanism calibration [5], and use the simplified models based on the redundant metrology paradigm.

However, such schemes should be used carefully for parallel mechanisms,

since joint control does not take into account the kinematic closures and may therefore yield high internal forces [8]. Indeed, a peculiarity of parallel mechanisms is that there always exists a joint velocity vector to realize a desired end-effector Cartesian velocity, but that there may exist configurations where there does not exist an end-effector Cartesian velocity associated to a given joint velocity vector. Moreover, there may exist several end-effector poses associated to a given joint configuration. Hence, a simple joint control may converge to the wrong end-effector pose, even though it converges to the correct joint configuration.

Visual servoing Second, vision can be additionally used to perform visual servoing [11]. Indeed, instead of measuring the end-effector pose and convert it into joint values, one could think of using this measure directly for control. Recall that there exist many visual servoing techniques ranging from position-based visual servoing (PBVS) [27] (when the pose measurement is explicit) to image-based visual servoing (IBVS) [11] (when it is made implicit by using only image measurements). Most applications embed the vision system onto the end-effector to position the latter with respect to a rigid object whose accurate position is unknown, but one can also find applications with a fixed camera observing the end-effector [15]. The interested reader is referred to [7] for a thorough and up-to-date state-of-the-art.

Visual servoing techniques are very effective since they close the control loop over the vision sensor. This yields a high robustness to disturbances as well as to calibration errors. Indeed, these errors only appear in a Jacobian matrix but not in the regulated error.

Essentially, visual servoing techniques generate a Cartesian desired velocity which is converted into joint velocities by the robot inverse differential kinematic model. Hence, one can translate such techniques to parallel mechanisms, as in [20, 19, 17] (for parallel robots with a reduced number of degrees of freedom), by observation of the robot end-effector and the use of standard kinematic models. It is rather easier than in the serial case, since the inverse differential kinematic model of a parallel mechanism is usually analytical. Moreover, for parallel mechanisms, since the joint velocities are filtered through the inverse differential kinematic model, they are admissible, in the sense that they do not generate internal forces. More precisely, this is only true in the ideal case. However, if the estimated inverse differential kinematic model used for control is close enough to the actual one, the joint velocities will be closely admissible, in the sense that they do not generate high internal forces. The only difficulty for end-effector visual servoing of a parallel mechanism would come from the dependency of the inverse differential kinematic model to the Cartesian pose, which would need be estimated, but, as stated above, vision can also do that [9, 21] ! Notice that this point pleads for PBVS rather than IBVS of parallel mechanisms, which is effectively the choice made in [20, 19, 17].

From the above discussion, we thus highly recommend to use visual servoing for parallel mechanism control.

A novel approach However, the previous two ways consist solely in a simple adaptation of now classical control schemes, which, although probably very efficient, are not very innovative. Moreover, the direct application of visual servoing techniques assumes implicitly that the robot inverse differential kinematic model is given and that it is calibrated. Therefore, modeling, identification and control have small interaction with each other. Indeed, the model is usually defined for control using proprioceptive sensing only and does not foresee the use of vision for control, then identification and control are defined later on with the constraints imposed by the model (Figure 2). This is useful for modularity but this might not be efficient in terms of accuracy as well as of experimental set-up time.

Figure 2: Usual cascade from modeling to vision-based control.

On the opposite, a unified framework for modeling, identification and control, apart from being definitely more satisfying for the mind, would certainly open a path to higher efficiency. Indeed, instead of having identification and control being driven by the initial modeling stage, one could have a model taking into account the use of vision for control and hence for identification. To do so, it is necessary to fuse robot kinematics and projective geometry into a projective kinematic model (Figure 3). Thus, we propose a novel third way to use vision, which gathers the advantages of redundant metrology and of visual servoing and avoids most of their drawbacks.

Moreover, observing the end-effector of a parallel mechanism by vision may be incompatible with its application. For instance, it is not wise to imagine observing the end-effector of a machining tool. On the opposite, it should not be a problem to observe the legs of the mechanism, even in such extreme cases. Thereby one would turn vision from an exteroceptive sensor to a somewhat more proprioceptive sensor. This brings us back to the redundant metrology paradigm.

Figure 3: Simplified cascade from modeling to vision-based control using a projective kinematic model.

Parallel mechanisms are most often designed with slim and rectilinear legs. Thus, one is inclined to consider them as straight lines as it was done for their kinematic analysis [30, 16] or kinematic calibration [5, 34, 33, 35]. Therefore, the line geometry [31, 36] is certainly the heart of the unification, all the more as line geometry is widely used in kinematic analysis [22, 32] and computer vision [12] and has already been used for visual servoing [23, 1, 24].

Previous work on kinematic calibration [34, 33, 35] already considered vision as a way to deliver contact-less metrological redundancy. However, to the exception of [35], the models that were calibrated remain classical. Indeed, vision was only used for sensing and neither modeling nor control was questioned from the vision point of view. A first step in this direction was made in [2] were vision was used already at the modeling stage in order to derive a visual servoing scheme based on the observation of a Gough-Stewart parallel robot [14, 37]. In that method, the legs orientation were chosen as visual primitives and control was derived based on their reconstruction from the image. Although this reconstruction step consists only in computing the intersection of the two cylinder edges in the image, it might not be very accurate for intrinsically geometrical reasons. Indeed, if a leg is parallel to the image, its edges will appear as parallel lines in the image and their intersection will lie at the infinite. Thus, in a close case to this one, the reconstruction will not be highly accurate and will impair the control.

In practice, this case is rapidly encountered. Indeed, since tracking lines in the image might be hard in the presence of camera distortion, one would chose a rather long focal lens (6 mm is here a long focal). Then, to observe all the legs in the image, one would place the camera at some distance from the manipulator (say, 1 m away from a desktop Gough-Stewart manipulator). In such an easy to

set-up case, the cylinder edges appear nearly parallel in the image and control becomes unstable.

Consequently, following the original idea and the common habit in visual servoing to derive control laws as close as possible to the image space, we propose in this paper to servo the leg edges rather than the leg orientation.

The outline of the paper is as follows. Section 2 recalls the model used for lines, then uses it to express the Gough-Stewart platform kinematics in the (static) camera frame and finally, recalls some useful geometric properties associated to the fact that most parallel mechanisms have cylindrical legs. Section 3 introduces, under the cylindrical legs assumption, the novel control law, expressed in the image and using the apparent edges of the legs as visual primitives for control. Section 4 presents simulation results validating the approach and the first ever experimental results of visual servoing using legs observation in its two variants: the already published leg orientation-based control and the novel edge-based control.

2 Modeling

2.1 Line modeling

A line $\mathcal{L}$ in space, expressed in the camera frame, is defined by its Binormalized Plücker coordinates [1]:

$$\mathcal{L} \equiv ({}^c\underline{\mathbf{u}}, {}^c\underline{\mathbf{n}}, {}^cn) \tag{1}$$

where ${}^c\underline{\mathbf{u}}$ is the unit vector giving the spatial orientation of the line, ${}^c\underline{\mathbf{n}}$ is the unit vector defining the so-called interpretation plane of line $\mathcal{L}$ and ${}^cn$ is a non-negative scalar. The latter are defined by ${}^cn{}^c\underline{\mathbf{n}} = {}^c\mathbf{P} \times {}^c\underline{\mathbf{u}}$ where ${}^c\mathbf{P}$ is any point on the line. Notice that, using this notation, the well-known (normalized) Plücker coordinates [31, 32] are the couple $({}^c\underline{\mathbf{u}}, {}^cn{}^c\underline{\mathbf{n}})$.

The projection of such a line in the image plane, expressed in the camera frame, has for characteristic equation:

$${}^c\underline{\mathbf{n}}^T{}^c\mathbf{p} = 0 \tag{2}$$

where ${}^c\mathbf{p}$ are the coordinates in the camera frame of a point in the image plane, lying on the line.

With the intrinsic parameters $\mathbf{K}$, one can obtain the line equation in pixel coordinates ${}^p\underline{\mathbf{n}}$ from:

$${}^p\underline{\mathbf{n}}^T{}^p\mathbf{p} = 0 \tag{3}$$

Indeed, replacing ${}^p\mathbf{p}$ by $\mathbf{K}{}^c\mathbf{p}$ in this expression yields:

$${}^p\underline{\mathbf{n}}^T\mathbf{K}{}^c\mathbf{p} = 0 \tag{4}$$

By identification of (2) and (4), one obtains

$$ {}^{p}\underline{\mathbf{n}} = \frac{\mathbf{K}^{-T}{}^{c}\underline{\mathbf{n}}}{\|\mathbf{K}^{-T}{}^{c}\underline{\mathbf{n}}\|} \tag{5} $$

$$ {}^{c}\underline{\mathbf{n}} = \frac{\mathbf{K}^{T}{}^{p}\underline{\mathbf{n}}}{\|\mathbf{K}^{T}{}^{p}\underline{\mathbf{n}}\|} \tag{6} $$

Notice that for numerical reasons, one should use normalized pixel coordinates. Namely, let us define the pixel frame by its origin located at the image center (*i.e.* the intersection of the image diagonals) and such that the pixel coordinates vary approximately between -1 and +1, according to the choice of the normalizing factor, which can be the image horizontal dimension in pixels, or its vertical dimension, or its diagonal.

2.2 Vision-based kinematics of an hexapod

Consider the hexapod in Figure 1. It has 6 cylindrical legs of varying length $q_i, i \in 1..6$, attached to the base by spherical joints located in points $\mathbf{A}_i$ and to the moving platform (end-effector) by spherical joints located in points $\mathbf{B}_i$.

Rather than using the standard scalar inverse kinematic model of such an hexapod given by

$$ \forall i \in 1..6, \quad q_i^2 = \overrightarrow{\mathbf{A}_i\mathbf{B}_i}^{T}\overrightarrow{\mathbf{A}_i\mathbf{B}_i} \tag{7} $$

expressing that q_i is the length of vector $\overrightarrow{\mathbf{A}_i\mathbf{B}_i}$, it is preferable for the subsequent derivation to use the vector form, introduced as the *vision-based kinematics of the hexapod* expressed in the camera frame in [2]:

$$ q_i\,{}^{c}\underline{\mathbf{u}}_i = {}^{c}\mathbf{R}_e\,{}^{e}\mathbf{B}_i + {}^{c}\mathbf{t}_e - {}^{c}\mathbf{A}_i \tag{8} $$

where ${}^{c}\underline{\mathbf{u}}_i$ is the spatial orientation or the ith leg. From the inverse kinematic model, one easily obtains the differential inverse kinematic model:

$$ \dot{\mathbf{q}} = {}^{c}\mathbf{J}_c^{inv}\,{}^{c}\tau_c \tag{9} $$

$$ {}^{c}\mathbf{J}_c^{inv} = -\begin{bmatrix} {}^{c}\underline{\mathbf{u}}_1^{T} & ({}^{c}\mathbf{A}_1 \times {}^{c}\underline{\mathbf{u}}_1)^{T} \\ \vdots & \vdots \\ {}^{c}\underline{\mathbf{u}}_6^{T} & ({}^{c}\mathbf{A}_6 \times {}^{c}\underline{\mathbf{u}}_6)^{T} \end{bmatrix} \tag{10} $$

where ${}^{c}\tau_c$ is the Cartesian velocity of the camera frame, considered as attached to the base frame and moving with respect to a fixed end-effector, expressed in itself and ${}^{c}\underline{\mathbf{u}}_i, i = 1..6$ are the unit vectors giving the pointing direction of each leg in the camera frame. Under the assumption that the legs are cylinders, those vectors can be easily detected as the intersection of the two cylinder edges in the image plane.

2.3 Cylindrical leg observation

It was shown in [4] that the edges of the ith cylindrical leg of the hexapod are given, in the camera frame, by

$$
\begin{aligned}
{}^c\underline{\mathbf{n}}_i^1 &= -\cos\theta_i {}^c\underline{\mathbf{h}}_i - \sin\theta_i {}^c\underline{\mathbf{u}}_i \times {}^c\underline{\mathbf{h}}_i && (11)\\
{}^c\underline{\mathbf{n}}_i^2 &= +\cos\theta_i {}^c\underline{\mathbf{h}}_i - \sin\theta_i {}^c\underline{\mathbf{u}}_i \times {}^c\underline{\mathbf{h}}_i && (12)
\end{aligned}
$$

where $\cos\theta_i = \sqrt{{}^ch_i^2 - R^2}/{}^ch_i$, $\sin\theta_i = R/{}^ch_i$ and $({}^c\underline{\mathbf{u}}_i, {}^c\underline{\mathbf{h}}_i, {}^ch_i)$ are the Binormalized Plücker coordinates of the ith cylinder axis and R is the cylinder radius.

It was also shown that the leg orientation, expressed in the camera frame, is given by

$$
{}^c\underline{\mathbf{u}}_i = \frac{{}^c\underline{\mathbf{n}}_i^1 \times {}^c\underline{\mathbf{n}}_i^2}{\|{}^c\underline{\mathbf{n}}_i^1 \times {}^c\underline{\mathbf{n}}_i^2\|} \tag{13}
$$

Notice that the geometric interpretation of this result is that ${}^c\underline{\mathbf{u}}_i$ is, up to a scale factor, the vanishing point of the two image edges, i.e. their intersection point in the image.

Let us remark now that each cylinder edge is a line in space, with Binormalized Plücker expressed in the camera frame $({}^c\underline{\mathbf{u}}_i, {}^c\underline{\mathbf{n}}_i^j, {}^cn_i^j)$. Moreover, the attachment point $\mathbf{A}_i$ is lying on the cylinder axis at distance R from the edge. Consequently, a cylinder edge is entirely defined by the following constraints, expressed here in the camera frame, although valid in any frame:

$$
\begin{aligned}
{}^c\mathbf{A}_i^T {}^c\underline{\mathbf{n}}_i^j &= -R && (14)\\
{}^c\underline{\mathbf{n}}_i^{jT} {}^c\underline{\mathbf{n}}_i^j &= 1 && (15)\\
{}^c\underline{\mathbf{u}}_i^T {}^c\underline{\mathbf{n}}_i^j &= 0 && (16)
\end{aligned}
$$

3 Edge-based visual servoing

In the following subsection, the i subscript denoting the leg number was removed for clarity sake.

3.1 Interaction matrix

The interaction matrix $\mathbf{N}^T$ relating the Cartesian velocity ${}^c\tau_c$ to the time derivative of the cylinder edges ${}^c\dot{\underline{\mathbf{n}}}^j$, expressed in the pixel frame:

$$
{}^c\dot{\underline{\mathbf{n}}}^j = \mathbf{N}^T {}^c\tau_c \tag{17}
$$

can be decomposed into the product of three matrices:

$$
\mathbf{N}^T = {}^p\mathbf{J}_c {}^h\mathbf{J}_u \mathbf{M}^T \tag{18}
$$

From right to left, the first one is the interaction matrix associated to the leg orientation, it thus relates the time derivative of a leg orientation to ${}^c\tau_c$.

The second transforms leg orientation velocities into leg edge velocities both expressed in the camera frame. Finally, the third one is associated to the camera-to-pixel change of frame. Below, the expression of the leg orientation interaction matrix is first recalled then the last two matrices are derived.

3.1.1 Leg orientation interaction matrix

The control proposed in [2] servoed the geodesic error between the current and desired legs orientation (${}^c\underline{\mathbf{u}} \times {}^c\underline{\mathbf{u}}^*$) and thus introduced the interaction matrix associated to a leg orientation ${}^c\underline{\mathbf{u}}$:

$$
\begin{aligned}
{}^c\dot{\underline{\mathbf{u}}} &= \mathbf{M}^T\, {}^c\tau_c && (19)\\
\mathbf{M}^T &= -\frac{1}{q}\left(\mathbf{I}_3 - {}^c\underline{\mathbf{u}}\,{}^c\underline{\mathbf{u}}^T\right)\begin{bmatrix}\mathbf{I}_3 & -[{}^c\mathbf{A} + q\,{}^c\underline{\mathbf{u}}]_\times\end{bmatrix} && (20)
\end{aligned}
$$

3.1.2 Edge velocity in the camera frame

Let us first derive the time derivative of a cylinder edge, expressed in the camera frame, and under the kinematic constraint that the cylinder is attached to the base by a spherical or universal joint located in $\mathbf{A}$. To do so, let us differentiate the constraints (14)-(16):

$$
\begin{aligned}
{}^c\dot{\underline{\mathbf{n}}}^{jT}\mathbf{A} &= 0 && (21)\\
{}^c\dot{\underline{\mathbf{n}}}^{jT}\,{}^c\underline{\mathbf{n}}^j &= 0 && (22)\\
{}^c\dot{\underline{\mathbf{n}}}^{jT}\,{}^c\underline{\mathbf{u}} + {}^c\underline{\mathbf{n}}^{jT}\,{}^c\dot{\underline{\mathbf{u}}} &= 0 && (23)
\end{aligned}
$$

From (22) and the fact that $(\underline{\mathbf{u}}, \underline{\mathbf{n}}, \underline{\mathbf{u}} \times \underline{\mathbf{n}})$ form an orthonormal basis [1], one can state:

$$
{}^c\dot{\underline{\mathbf{n}}}^j = \alpha\,{}^c\underline{\mathbf{u}} + \beta\,{}^c\underline{\mathbf{u}} \times {}^c\underline{\mathbf{n}}^j \qquad (24)
$$

Inserting this expression into (21) and (23) yields

$$
\alpha = -{}^c\underline{\mathbf{n}}^{jT}\,{}^c\dot{\underline{\mathbf{u}}}\,, \quad \beta = \frac{{}^c\mathbf{A}^T\,{}^c\underline{\mathbf{u}}}{{}^c\mathbf{A}^T({}^c\underline{\mathbf{u}} \times {}^c\underline{\mathbf{n}}^j)}\,{}^c\underline{\mathbf{n}}^{jT}\,{}^c\dot{\underline{\mathbf{u}}} \qquad (25)
$$

Consequently, one obtains the relationship between the time derivative of a leg edge, expressed in the camera frame, and the time derivative of the leg orientation

$$
\begin{aligned}
{}^c\dot{\underline{\mathbf{n}}}^j &= {}^h\mathbf{J}_u\,{}^c\dot{\underline{\mathbf{u}}} \qquad (= {}^h\mathbf{J}_u\mathbf{M}^T\,{}^c\tau_c) && (26)\\
{}^h\mathbf{J}_u &= \left(\frac{{}^c\mathbf{A}^T\,{}^c\underline{\mathbf{u}}}{{}^c\mathbf{A}^T({}^c\underline{\mathbf{u}} \times {}^c\underline{\mathbf{n}}^j)}({}^c\underline{\mathbf{u}} \times {}^c\underline{\mathbf{n}}^j) - {}^c\underline{\mathbf{u}}\right){}^c\underline{\mathbf{n}}^{jT} && (27)
\end{aligned}
$$

3.1.3 Image line velocity in pixel coordinates

Let us now derive the Jacobian associated to the change of frame, where the time derivative of an image line is expressed, from the camera frame to the pixel

frame. Note that this paragraph holds for any image line, not only for cylinder edges.

Rewriting (5) as

$$ {}^p\underline{\mathbf{n}} = \mu({}^c\underline{\mathbf{n}})\mathbf{K}^{-T}{}^c\underline{\mathbf{n}} \tag{28} $$

we can differentiate the latter with time:

$$ {}^p\dot{\underline{\mathbf{n}}} = \frac{d\mu({}^c\underline{\mathbf{n}})}{dt}\mathbf{K}^{-T}{}^c\underline{\mathbf{n}} + \mu({}^c\underline{\mathbf{n}})\mathbf{K}^{-T}{}^c\dot{\underline{\mathbf{n}}} \tag{29} $$

Taking into account again that ${}^p\underline{\mathbf{n}}$ is a unit vector (22), one gets

$$ \left(\frac{d\mu({}^c\underline{\mathbf{n}})}{dt}\mathbf{K}^{-T}{}^c\underline{\mathbf{n}}\right)^T {}^p\underline{\mathbf{n}} + \mu({}^c\underline{\mathbf{n}}){}^p\underline{\mathbf{n}}^T\mathbf{K}^{-T}{}^c\dot{\underline{\mathbf{n}}} = 0 \tag{30} $$

Using (5) again, this simplifies into

$$ \frac{d\mu({}^c\underline{\mathbf{n}})}{dt} = -\mu({}^c\underline{\mathbf{n}})^2{}^p\underline{\mathbf{n}}^T\mathbf{K}^{-T}{}^c\dot{\underline{\mathbf{n}}} \tag{31} $$

Inserting this result in (29) yields

$$ {}^p\dot{\underline{\mathbf{n}}} = \left(-\mathbf{K}^{-T}{}^c\underline{\mathbf{n}}^T\mu({}^c\underline{\mathbf{n}})^2{}^p\underline{\mathbf{n}}^T + \mu({}^c\underline{\mathbf{n}})\mathbf{I}_3\right)\mathbf{K}^{-T}{}^c\dot{\underline{\mathbf{n}}} \tag{32} $$

which simplifies into

$$ {}^p\dot{\underline{\mathbf{n}}} = \mu({}^c\underline{\mathbf{n}})\left(\mathbf{I}_3 - {}^p\underline{\mathbf{n}}{}^p\underline{\mathbf{n}}^T\right)\mathbf{K}^{-T}{}^c\dot{\underline{\mathbf{n}}} \tag{33} $$

Introducing (6) in (28) proves that

$$ \mu({}^c\underline{\mathbf{n}}) = \|\mathbf{K}^T{}^p\underline{\mathbf{n}}\| \tag{34} $$

from which we finally obtain the relationship between the time derivative of a line, expressed in the image frame, and the same expressed in the camera frame

$$ {}^p\dot{\underline{\mathbf{n}}} = {}^p\mathbf{J}_c{}^c\dot{\underline{\mathbf{n}}} \qquad (= {}^p\mathbf{J}_c{}^h\mathbf{J}_u\mathbf{M}^T{}^c\tau_c) \tag{35} $$

$$ {}^p\mathbf{J}_c = \|\mathbf{K}^T{}^p\underline{\mathbf{n}}\|\left(\mathbf{I}_3 - {}^p\underline{\mathbf{n}}{}^p\underline{\mathbf{n}}^T\right)\mathbf{K}^{-T} \tag{36} $$

thus proving (18).

3.2 Control

Since we want to drive the unit vectors associated to the leg edges to their desired values, we choose to servo the geodesic errors

$$ \mathbf{e}_{i,j} = {}^p\underline{\mathbf{n}}_i^j \times {}^p\underline{\mathbf{n}}_i^{j*}, j = 1..2, i = 1..6 \tag{37} $$

the time derivative of which is

$$ \dot{\mathbf{e}}_{i,j} = \mathbf{L}_{i,j}^T{}^p\dot{\underline{\mathbf{n}}}_i^j \tag{38} $$

$$ \mathbf{L}_{i,j}^T = -[{}^p\underline{\mathbf{n}}_i^{j*}]_\times\mathbf{N}_{i,j}^T \tag{39} $$

where $i = 1..6$ denotes the legs and $j = 1..2$ the edges.

Now, the standard method applies: we stack each individual errors in a single over-constrained vector $\mathbf{e}$ and each associated individual interaction matrices $\mathbf{L}_{i,j}^T$ into a compound one $\mathbf{L}^T$ and impose a first-order convergence to $\mathbf{e}$. This yields the following pseudo-control vector ${}^c\tau_c$

$$ {}^c\tau_c = -\lambda \mathbf{L}^T \mathbf{e} \tag{40} $$

which is fed to the actuators through the vision-based differential inverse kinematic model (10) to deliver the final control signal

$$ \dot{\mathbf{q}} = -\lambda {}^c\mathbf{J}_c^{inv} \mathbf{L}^T \mathbf{e} \tag{41} $$

Notice that this control makes use of the detected edges in the image, the joint values, the intrinsic parameters and the attachment points of the legs onto the base expressed in the camera frame. However, notice that neither the attachment points of the legs onto the mobile platform nor the radius of the legs are used here explicitly, which reduces the number of kinematic parameters to be calibrated. We strongly suspect that, if needed, we can also get rid of the joint values as in [3] by using their median value, but this is not the matter of this paper.

4 Results

4.1 Experimental set-up

Experiments were performed on a commercial DeltaLab hexapod, such that ${}^b\mathbf{A}_{2k} = R_b \begin{pmatrix} \cos(k\frac{\pi}{3}+\alpha) \\ \sin(k\frac{\pi}{3}+\alpha) \\ 0 \end{pmatrix}$, ${}^b\mathbf{A}_{2k+1} = R_b \begin{pmatrix} \cos(k\frac{\pi}{3}-\alpha) \\ \sin(k\frac{\pi}{3}-\alpha) \\ 0 \end{pmatrix}$, ${}^e\mathbf{B}_{2k} = R_e \begin{pmatrix} \cos(k\frac{\pi}{3}+\beta) \\ \sin(k\frac{\pi}{3}+\beta) \\ 0 \end{pmatrix}$, ${}^e\mathbf{B}_{2k+1} = R_e \begin{pmatrix} \cos(k\frac{\pi}{3}-\beta) \\ \sin(k\frac{\pi}{3}-\beta) \\ 0 \end{pmatrix}$, $k \in \{0,1,2\}$ with $R_b = 270mm$, $\alpha = 4.25°$, $R_e = 195mm$, $\beta = 5.885°$ and the legs range are $[345mm, 485mm]$.

The robot is observed by a IEEE 1394 digital Sony camera with 4.8mm focal length, placed in front of it. Figure 4 (left) presents a schematic view of the initial (solid line) and desired (dotted line) configurations of the robot, while Figure 4 (right) is obtained by merging the (video inverted) view from the camera of the robot in the initial configuration and the view in the desired configuration. The initial configuration yields equal leg lengths (375mm) whereas in the desired configuration the two legs in the back are extended by an additional 100mm.

To show the robustness of the approach, we deliberately placed ourselves in a difficult configuration: approximate calibration of the robot and camera, camera placed 70cm away from the robot base center, two legs close to be in a frontal parallel configuration.

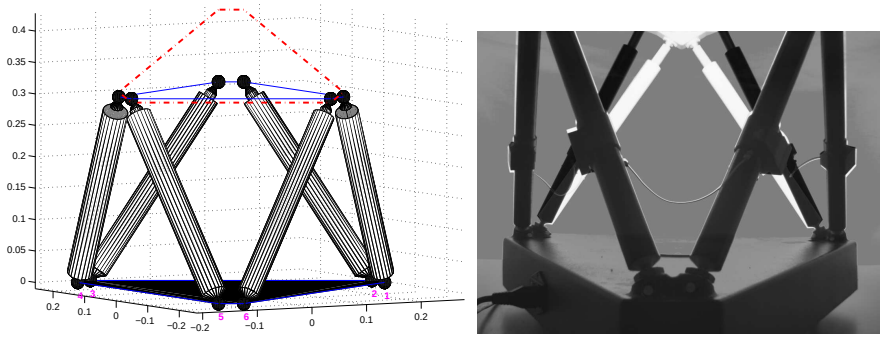

Figure 4: Composition of the desired (black) and initial (white) configurations.

4.2 Simulation

Here, we compare the behaviors of orientation-based control and edge-based control in the perfect noise-free case, where calibration is perfect and the camera is placed so that it preserves the vertical symmetry observed in Figure 4 (right). Of course, in both cases, the controlled errors converge exponentially to zero. It shall be mentioned that the non-controlled errors (*i.e.* orientation errors in edge-based control and edge errors in orientation-based control) also converge exponentially to zero, although with a slightly different rate. For sake of conciseness, these results are not displayed here, since they are somehow included in the experimental results below.

Nevertheless, it is worth to have a look at the spatial trajectory of the end-effector under both controls. Indeed, the symmetry is preserved by both control: the end-effector moves in the vertical plane passing through the robot base center and the center of projection. However, as seen in Figure 5, the trajectory in this plane is rectilinear in the case of orientation-based control while edge-based control yields a slightly bent trajectory.

4.3 Experiments

The experimental robot has an analog joint position-controller that we interfaced with Linux-RTAI. Joint velocity control is emulated through this position-controller with an approximate[1] 20ms sampling period. Frame grabbing, line tracking and numerical computation are performed using ViSP, an open C++ library for visual servoing [25].

It also has to be noticed that the mechanism presents high frictious disturbances that have not yet been compensated for since friction seems to depend non trivially on the robot configuration. Hence, to overcome these disturbances, we implemented the visual servoing control with an adaptive gain, function of

[1]This part is not yet running under RTAI, but only under standard Linux.

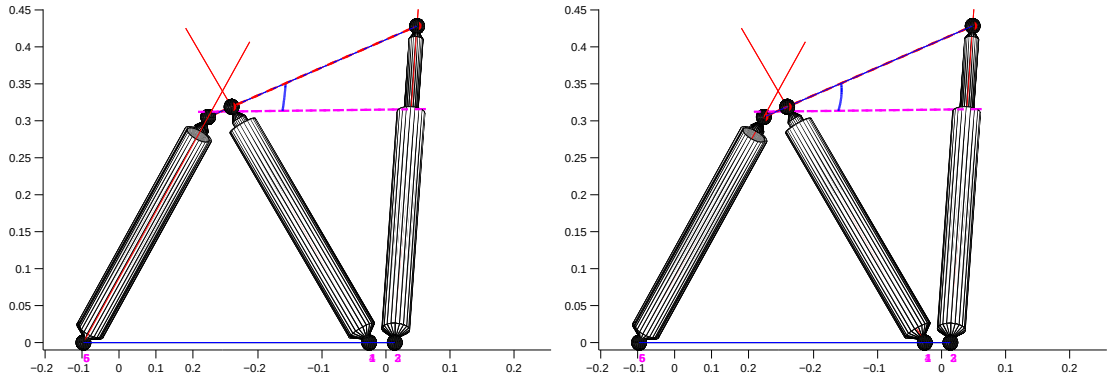

Figure 5: Spatial trajectory (side-view of the platform) of the end-effector under orientation-based control (left) and under edge-based control (right).

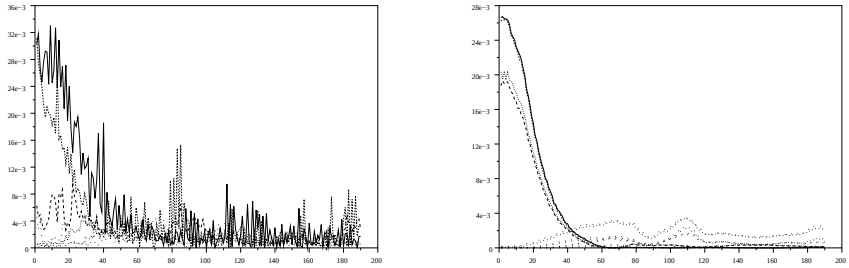

Figure 6: Behavior of orientation-based control: Evolution of the controlled leg orientation errors (left) and of the non-controlled edge errors (right) with respect to time.

the controlled error norm: low at init, high near convergence.

We display for both controls the evolution of both the controlled and non-controlled errors: in Figure 6 for orientation-based control and in Figure 7 for edge-based control. The exponential behavior is found again, yet disturbed by friction and distorted by the adaptive gain strategy.

It is noticeable (Figure 6) that the orientation error signal is extremely noisy, as expected. Thus, orientation-based control tries its best to servo it to zero, but does not succeed in bringing the robot to the desired configuration since the edge errors do not reach zero. On the opposite (Figure 7), edge-based control succeeds (the edges are almost aligned on their reference) even though the orientation signal is noisy and biased. Figure 8 presents a complementary view of this behavior by displaying the evolution of both the controlled and non controlled error norms: the orientation signal is very noisy and converges equally poorly in both controls (left) while the edge signal is much cleaner and

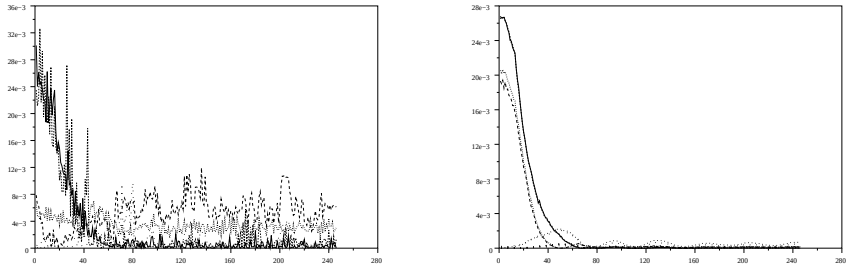

Figure 7: Behavior of edge-based control: Evolution of the non-controlled leg orientation errors (left) and of the controlled leg edge errors (right) with respect to time.

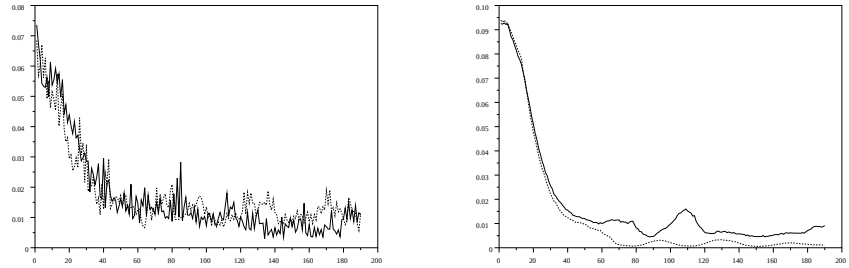

Figure 8: Comparison of the behavior of orientation-based (solid line) and edge-based control (dotted line): evolution of the norm of the legs orientation error (left) and of the edges error (right) with respect to time.

converges only in edge-based control.

The reason for such a behavior is, as stated in the early introduction, that noise is appearing in the servoed error in orientation-based control while it only appears in the interaction matrix in edge-based control.

5 Conclusion

We extended previous results concerning (PBVS-like) leg orientation-based control of a Gough-Stewart to an (IBVS-like) edge-based visual servoing scheme. It benefits from the advantages of the orientation-based visual servoing of the Gough-Stewart legs: reduced calibration parameters set, low dependence on the joint values and ability to servo the robot even though the end-effector is not visible. However, we improved the practical robustness (although it still has to be proven theoretically) by servoing the legs in the image: almost all the cal-

ibration parameters (intrinsic parameters of the camera and base points) and numerical errors remain located in the interaction matrix.

To do so, we took advantage of a common use of line geometry in kinematics, vision and visual servoing. This allows for an optimal modeling of Gough-Stewart parallel robots, provided that vision is used at control time. This modeling was established under the hypothesis that the camera is calibrating, but this result might be extendable to the use of an uncalibrated camera. Nevertheless, this extension is not necessary since the control is done in the very projective space associated to image lines, while the reconstructed or calibrated Euclidean terms only appear in the interaction matrix where extreme accuracy is not required.

However, self-occlusions of the mechanism with respect to a single camera are still a matter of study, although the observation of edges should simplify the problem since the two edges of a given leg are seldom hidden simultaneously. A way to overcome the occlusion problem is to turn oneself to multi-camera perception systems.

Acknowledgment

This study was jointly funded by CPER Auvergne 2003-2005 program and by the CNRS-ROBEA program through the MP2 project. The authors warmly thank Eric Marchand for providing them with an early version of ViSP.

References

[1] N. Andreff, B. Espiau, and R. Horaud. Visual servoing from lines. *Int. Journal of Robotics Research*, 21(8):679–700, August 2002.

[2] N. Andreff, A. Marchadier, and P. Martinet. Vision-based control of a Gough-Stewart parallel mechanism using legs observation. In *Proc. Int. Conf. Robotics and Automation (ICRA'05)*, pages 2546–2551, Barcelona, Spain, May 2005.

[3] N. Andreff and P. Martinet. Visual servoing of a Gough-Stewart parallel robot without proprioceptive sensors. In *Fifth International Workshop on Robot Motion and Control (RoMoCo'05)*, Dymaczewo, Poland, June 23-25 2005.

[4] N. Andreff and P. Martinet. Visually servoing a gough-stewart parallel robot allows for reduced and linear kinematic calibration. In *Proc. Int. Conf. on Informatics in Control, Automation and Robotics (ICINCO'05)*, volume 3, pages 119–124, Barcelona, Spain, September 14–17 2005.

[5] N. Andreff, P. Renaud, P. Martinet, and F. Pierrot. Vision-based kinematic calibration of an H4 parallel mechanism: practical accuracies. *Industrial Robot: An international journal*, 31(3):273–283, May 2004.

[6] L. Baron and J. Angeles. The on-line direct kinematics of parallel manipulators under joint-sensor redundancy. In *Advances in Robot Kinematics*, pages 126–137. Kluwer Academic Publishers, 1998.

[7] H.I. Christensen and P. Corke, editors. *Int. Journal of Robotics Research – Special Issue on Visual Servoing*, volume 22, October 2003.

[8] B. Dasgupta and T.S. Mruthyunjaya. Force redundancy in parallel manipulators: theoretical and practical issues. *Mech. Mach. Theory*, 33(6):727–742, 1998.

[9] D. DeMenthon and L. Davis. Model-based object pose in 25 lines of code. *Int. Journal on Cmputer Vision*, 15(1/2):123–141, June 1995.

[10] P. Dietmaier. The Stewart-Gough platform of general geometry can have 40 real postures. In J. Lenarčič and M. L. Husty, editors, *Advances in Robot Kinematics: Analysis and Control*, pages 1–10. Kluwer, 1998.

[11] B. Espiau, F. Chaumette, and P. Rives. A New Approach To Visual Servoing in Robotics. *IEEE Trans. on Robotics and Automation*, 8(3), June 1992.

[12] O. Faugeras. *Three-Dimensional Computer Vision - A Geometric Viewpoint*. Artificial intelligence. MIT Press, Cambridge, MA, 1993.

[13] G. Gogu. Fully-isotropic T3R1-type parallel manipulator. In J. Lenarčič and C. Galletti, editors, *On Advances in Robot Kinematics*, pages 265–272. Kluwer Academic Publishers, 2004.

[14] V.E. Gough and S.G. Whitehall. Universal tyre test machine. In *Proc. FISITA 9th Int. Technical Congress*, pages 117–137, May 1962.

[15] R. Horaud, F. Dornaika, and B. Espiau. Visually Guided Object Grasping. *IEEE Transactions on Robotics and Automation*, 14(4):525–532, 1998.

[16] M. Husty. An algorithm for solving the direct kinematics of general Gough-Stewart platforms. *Mech. Mach. Theory*, 31(4):365–380, 1996.

[17] P. Kallio, Q. Zhou, and H. N. Koivo. Three-dimensional position control of a parallel micromanipulator using visual servoing. In Bradley J. Nelson and Editors Jean-Marc Breguet, editors, *Microrobotics and Microassembly II, Proceedings of SPIE*, volume 4194, pages 103–111, Boston, USA, November 2000.

[18] H.S. Kim and L.-W. Tsai. Evaluation of a Cartesian parallel manipulator. In J. Lenarčič and F. Thomas, editors, *Advances in Robot Kinematics: Theory and Applications*. Kluwer Academic Publishers, June 2002.

[19] H. Kino, C.C. Cheah, S. Yabe, S. Kawamua, and S. Arimoto. A motion control scheme in task oriented coordinates and its robustness for parallel wire driven systems. In *Int. Conf. Advanced Robotics (ICAR'99)*, pages 545–550, Tokyo, Japan, Oct. 25-27 1999.

[20] M.L. Koreichi, S. Babaci, F. Chaumette, G. Fried, and J. Pontnau. Visual servo control of a parallel manipulator for assembly tasks. In *6th Int. Symposium on Intelligent Robotic Systems, SIRS'98*, pages 109–116, Edimburg, Scotland, July 1998.

[21] JM. Lavest, M. Viala, and M. Dhome. Do we really need an accurate calibration pattern to achieve a reliable camera calibration. In *Proceedings of ECCV98*, pages 158–174, Freiburg, Germany, June 1998.

[22] J. M. MacCarthy. *Introduction to Theoretical Kinematics*. MIT Press, 1990.

[23] R. Mahony and T. Hamel. Visual servoing using linear features for under-actuated rigid-body dynamics. In *IEEE/RSJ Int. Conf. on Intelligent Robots Systems*, pages 1153–1158, Hawaii, USA, 2001.

[24] R. Mahony and T. Hamel. Image-based visual servo control of aerial robotic systems using linear image features. *IEEE Transactions on Robotics*, 21(2):227–239, April 2005.

[25] E. Marchand, F. Spindler, and F. Chaumette. ViSP: a generic software platform for visual servoing. *IEEE Robotics and Automation Magazine*, 12(4), December 2005.

[26] F. Marquet. *Contribution à l'étude de l'apport de la redondance en robotique parallèle*. PhD thesis, LIRMM - Univ. Montpellier II, October 2002.

[27] P. Martinet, J. Gallice, and D. Khadraoui. Vision based control law using 3d visual features. In *Proc. World Automation Congress, WAC'96, Robotics and Manufacturing Systems*, volume 3, pages 497–502, Montpellier, France, May 1996.

[28] J-P. Merlet. An algorithm for the forward kinematics of general 6 d.o.f. parallel manipulators. Technical Report 1331, INRIA, November 1990.

[29] J.P. Merlet. http://www-sop.inria.fr/coprin/equipe/merlet.

[30] J.P. Merlet. *Parallel robots*. Kluwer Academic Publishers, 2000.

[31] J. Plücker. On a new geometry of space. *Philosophical Transactions of the Royal Society of London*, 155:725–791, 1865.

[32] H. Pottmann, M. Peternell, and B. Ravani. Approximation in line space – applications in robot kinematics and surface reconstruction. In J. Lenarčič and M. L. Husty, editors, *Advances in Robot Kinematics: Analysis and Control*, pages 403–412. Kluwer Academic Publishers, 1998.

[33] P. Renaud, N. Andreff, G. Gogu, and Ph. Martinet. On vision-based kinematic calibration of a Stewart-Gough platform. In *11th World Congress in Mechanism and Machine Science (IFToMM2003)*, pages 1906–1911, Tianjin, China, April 1-4 2004.

[34] P. Renaud, N. Andreff, Ph. Martinet, and G. Gogu. Kinematic calibration of parallel mechanisms: A novel approach using legs observation. *IEEE Transactions on Robotics*, 2005. In press.

[35] P. Renaud, N. Andreff, F. Pierrot, and P. Martinet. Combining end-effector and legs observation for kinematic calibration of parallel mechanisms. In *IEEE Int. Conf. on Robotics and Automation (ICRA 2004)*, pages 4116–4121, New Orleans, USA, May 2004.

[36] J.G. Semple and G.T. Kneebone. *Algebraic Projective Geometry*. Oxford Science Publication, 1952.

[37] D. Stewart. A platform with six degrees of freedom. In *Proc. IMechE (London)*, volume 180, pages 371–386, 1965.

[38] J. Wang and O. Masory. On the accuracy of a Stewart platform - Part I: The effect of manufacturing tolerances. In *IEEE Int. Conf. on Robotics and Automation (ICRA 1993)*, pages 114–120, 1993.

[39] X. Zhao and S. Peng. Direct displacement analysis of parallel manipulators. *Journal of Robotics Systems*, 17(6):341–345, 2000.